Linux für Studenten

Michael Kofler
Jürgen Plate

4. Dezember 2007

Inhaltsverzeichnis

Vorwort

In den letzten Jahren hat Linux viel Bewegung in die Welt der Betriebssysteme gebracht. Zwar nimmt Microsoft mit Windows nach wie vor eine dominierende Rolle ein. Aber Linux ist drauf und dran, Platz zwei zu erobern. Immer mehr Firmen-Server laufen unter Linux – selbst wenn andere Rechner im Netzwerk noch unter Windows laufen. Immer mehr Unternehmen und Organisationen erwägen, auch Büroarbeitsplätze auf Linux umzustellen. (Die Stadt München ist ein prominentes Beispiel dafür.)

Der Erfolg von Linux hat viele Gründe, von denen hier nur die zwei wichtigsten genannt seien:

- Linux ist frei verfügbar. „Frei“ bezieht sich dabei nicht nur auf den Preis (Linux darf bei Einhaltung bestimmter Spielregeln kostenlos weitergegeben werden), sondern vor allem auf die Verfügbarkeit des Quellcodes: Wenn Sie mit Linux nicht zufrieden sind, können Sie (entsprechende Programmierkenntnisse vorausgesetzt) gleich selbst versuchen, das Problem zu lösen.
- Linux hat sich in den vergangenen Jahren wiederholt als stabiler und sicherer als Windows erwiesen. Während Windows-Anwender unter ständig neuen Virenangriffen stöhnen und verzweifelt ein Update nach dem anderen zu installieren versuchen, gibt es unter Linux vergleichsweise wenige Sicherheitsprobleme.
- Linux bietet den grossen Vorteil, dass sich jede Anwendung auch so einrichten lässt, dass sie automatisch und umbeaufsichtigt laufen kann. Was man mehr als einmal macht, kann man gleich den Rechner machen lassen.
- Da bei Linux generell auch alle Serverprozesse laufen können, lassen sich mit wenig Aufwand alle Kommandozeilen-Anwendungen auch mit einem Web-Frontend versehen und so benutzerfreundlich gestalten.

Linux hat aber auch seine Haken – insbesondere, wenn Sie den Komfort von Microsoft Windows oder des Mac OS gewohnt sind. Deshalb hat es sich auf dem Desktop auch noch nicht so stark verbreitet wie im Serverbereich:

- Die Bedienung von Linux ist nicht immer einfach, insbesondere bei der Nutzung der Kommandozeile.
- Linux unterstützt nicht sofort jede Hardware. Besonders ganz neue Hardware-Komponenten bereiten oft Schwierigkeiten.
- Für jedes Problem und für jede Anwendung gibt es bei Linux mindestens drei Lösungen, von denen der Einsteiger aber keine kennt.
- Die meisten Programme werden ständig weiterentwickelt. Deshalb werden einige Programme wesentlich mehr können als wir hier beschreiben.
- Es gibt zwar etliche, teilweise auch sehr gute Spiele, aber Linux ist kein System für Hardcore-Gamer.

Gerade die Vielfalt von Linux-Anwendungen und -Distributionen hält vielleicht den einen oder anderen davon ab, umzusteigen – auch, wenn man nach dem Sprung ins kalte Wasser ein superstabiles und preiswertes System bekommt. Dieses Buch will Ihnen einen Weg zum heimischen Linux-Rechner bahnen. Dabei sollen Sie Ihr Windows oder Mac OS überhaupt nicht aufgeben. Es gibt nun mal Anwendungen, die nur unter einem bestimmten Betriebssystem laufen. Wir glauben aber, dass Linux viele gleichwertige oder sogar bessere Tools kennt – mal ganz abgesehen von den Features, die dieses Betriebssystem sonst noch bietet. Und nicht zuletzt: Wenn Sie keine Lust haben, schon bei der Installation von Software alle möglichen persönlichen Daten zur Registrierung an irgendwen in Übersee zu versenden, ist Linux das richtige System für Sie.

Manchmal ist Ihnen vielleicht auch dieses Buch unbequem: Es gibt hier keine Sammlungen von Screenshots mit Anweisungen der Art *Klicken Sie hier* finden – nach dem Motto „300 Screenshots und 300 Zeilen Text geben ein Buch“ Unser Anliegen ist es, Ihnen die Philosophie von Unix/Linux nahe zu bringen – gewissermaßen *the Linux way to do it.* Mit diesem Buch werde Sie Linux nicht nur anwenden, sondern auch verstehen.

Damit keine falschen Erwartungen geweckt werden: Auf etwa 400 Seiten kann kein Mensch auf jeden Aspekt von Linux eingehen. Dazu würden selbst 30 Bücher dieses Umfangs nicht ausreichen! Deshalb gibt es viele Verweise auf die Linux-Dokumentation, die Dokumentation der einzelnen Programme (wobei wir gestehen müssen, dass die Entwickler manchmal etwas „schreibfaul“ sind), den „großen Bruder“ dieses Buchs (Michael Kofler: „Linux“, Addison-Wesley) und diverse Webseiten. Um das Nachlesen, Ausprobieren und Lernen kommen Sie also nicht herum (aber das Problem haben Sie auch bei anderen Betriebssystemen). Wenn Sie erst mal Blut geleckt haben, werden Sie von ganz alleine zum Linux-Crack (Wer erinnert sich noch an die Zeiten, wo man beim Commodore C64 jedes Byte des Betriebssystems persönlich kannte?)

Linux steht in einer Vielzahl von Distributionen zur Verfügung (z.B. Debian, Fedora, Gentoo, Mandravia, Red Hat, Slackware und SUSE). Natürlich sind

diese Distributionen einander recht ähnlich: Bei jeder erhalten Sie nach der Installation ein Linux-System, das in diesem Buch beschriebenen Programme zur Verfügung stellt. Es gibt aber auch beträchtliche Unterschiede, was die Administration und Konfiguration betrifft. Dieses Buch versucht, so weit das möglich ist, Linux unabhängig von irgendwelchen Distributionen zu beschreiben und gleichsam den gemeinsamen Nenner aller oder zumindest der meisten Linux-Distributionen zu finden. Wo das nicht möglich ist, beschränken sich die Detailinformationen auf die Debian-Distribution, die Sie auch auf den beiliegenden DVDs finden.

Das Buch richtet sich zwar an Studierende aller Fachrichtungen, die den Computer für Ihre Arbeit effizient nutzen wollen. Aber obwohl im Titel „... für Studenten steht“ (wobei natürlich die Studentinnen ebenso angesprochen werden, aber „Student-innen“ bekommen höchstens angehende Mediziner in der Pathologie zu sehen), kann auch jeder andere vom Inhalt profitieren, wenn die Zielrichtung in etwa ähnlich ist. Spezielle Vorkenntnisse werden nicht vorausgesetzt, der Leser sollte aber mit der Bedienung und Grundlagen der Arbeitsweise eines Computers vertraut sein – also kein reiner „Mäuseschubser“. Eine gewisse Lernbereitschaft und Experimentierfreude setzen wir voraus.

Um Enttäuschungen zu vermeiden, sind gleich an dieser Stelle einige Dinge aufgelistet, die Sie in diesem Buch nicht finden werden:

- **Hardware:** Dieses Buch beschreibt nur Hardware, soweit sie ein Wald-Feld-und-Wiesen-PC bietet. Wenn Sie spezielle (Multimedia-)Hardware unter Linux in Betrieb nehmen möchten, bleibt Ihnen ein Blick in die zu Ihrer Distribution verfügbaren Informationen sowie in die Online-Dokumentation nicht erspart.
- **Netzwerk- und Server-Administration:** Das Buch beschreibt Netzwerke undInternet primär aus dem Blickwinkel des Privatanwenders, der lediglich seinen eigenen Rechner an das Internet anbindet bzw. ein kleines Netzwerk verwaltet. Wenn Sie Linux dagegen als Intranet- oder Internet-Server in großen Netzen einsetzen möchten, benötigen Sie weitergehende Literatur.
- **Multimedia-Anwendungen:** Das Buch geht zwar auf wichtige Multimedia-Themen ein (Audio und Video, CDs und DVDs brennen), es gäbe zu diesem Thema aber noch viel mehr zu sagen.

Bedanken möchten wir uns an dieser Stelle
bei Linus Torvalds, der den Stein (Kern) zum Rollen brachte,
bei Tausenden Entwicklern, die Linux zu einem tollen Gesamtsystem machten,
bei den Debian-Entwicklern für die DVD zum Buch,
bei den Lektoren und Mitarbeitern des Pearson-Verlags für viel Geduld,
und bei Prof. Dr. Abdul Nachtigaller (Zamonien) für seine wertvollen Anregungen.

Michael Kofler und Jürgen Plate, April 2006
http://www.kofler.cc und http://www.netzmafia.de

Formales

In diesem Buch werden häufig Kommandos angegeben. Dabei werden die Teile, die tatsächlich einzugeben sind, fett hervorgehoben. Der Rest der Listings besteht aus dem so genannten Kommandoprompt (der systemabhängig ist) und Ausgaben des Kommandos.

Dazu gleich ein Beispiel: Bei den Zeichen `user$` handelt es sich um den Kommandoprompt. (Der Prompt sind die Zeichen, die im Kommandofenster vor jeder Eingabe angezeigt werden. Sie dürfen nicht eingegeben werden!) Dem folgt das eigentliche Kommando und in den weiteren Zeilen das Ergebnis der Ausgabe. (`ls` zeigt die Liste aller Dateien im aktuellen Verzeichnis an.)

```
user$  ls *.tex
preface.tex
kap1.tex
kap2.tex
...
```

Manche Kommandos können nur vom Systemadministrator `root` eingegeben werden. In diesem Fall wird der Kommandoprompt in diesem Buch als `root#` dargestellt:

```
root#  /etc/init.d/nfs restart
```

Falls einzelne Kommandos so lang sind, dass sie nicht in einer Zeile Platz finden, werden sie mit dem Zeichen \ auf zwei oder mehr Zeilen verteilt. In diesem Fall können Sie die Eingabe entweder in einer Zeile ohne \ tippen, oder wie im Buch auf mehrere Zeilen verteilen. \ ist also ein unter Linux zulässiges Zeichen, um mehrzeilige Kommandoeingaben durchzuführen.

Manchmal endet ein Satz mit einer Kommandoausgabe; es müßte daher korrekterweise ein Punkt am Ende des Kommandos oder der Ausgabe stehen. Da bei Unix der Punkt jedoch auch häufig in Dateinamen vorkommt, könnte er in diesem Fall zu einer Fehlinterpretation führen (insbesondere, wenn man den Satz gar nicht gelesen hat, sondern nur das Kommando abtippt). Deshalb lassen wir den Punkt in solchen Fällen weg (alle Deutschlehrer mögen uns vergeben).

Kapitel 1

Was ist Linux?

Einführung

In diesem Kapitel wollen wir Ihnen einen knappen Überblick über die Merkmale von Linux und die verfügbaren Programme geben und zeigen, wie weit die Anwendungsmöglichkeiten von Linux reichen. Wir setzen dabei voraus, dass Ihnen der Begriff „Betriebssystem“ nicht fremd ist.

1.1 Einführung

Linux ist ein Unix-ähnliches Betriebssystem. Der wichtigste Unterschied gegenüber herkömmlichen Unix-Systemen besteht darin, dass Linux zusammen mit dem vollständigen Quellcode frei weitergegeben werden darf. Details zu den Bedingungen, unter denen Linux und die dazugehörigen Programme weitergegeben werden dürfen, folgen an Ende dieses Kapitels. Schon lange vorher gab es **Unix**. Unix gehört also historisch gesehen zu den älteren Betriebssystemen. Dennoch ist es gleichzeitig ein modernes Betriebssystem, das von Anfang an mit Merkmalen ausgestattet war, die von anderen Systemen erst viel später in einer vergleichbaren Form angeboten wurden. Unter Unix gab es von Anfang an echtes Multitasking, eine Trennung der Prozesse voneinander (d. h. hohe Stabilität), klar definierte Zugriffsrechte für Dateien (d. h. hohe Sicherheit im Multiuser-Betrieb), ausgereifte Netzwerkfunktionen etc. Allerdings bot Unix vor ein bis zwei Jahrzehnten nur eine kommandozeilenorientierte Benutzeroberfläche und stellte hohe Hardware-Anforderungen. Deshalb wurde es fast ausschließlich auf teuren Workstations im wissenschaftlichen und industriellen Bereich eingesetzt.

Linux ist im Prinzip nichts anderes als eine neue Unix-Variante. Zu den Besonderheiten von Linux zählen die freie Verfügbarkeit des gesamten Quelltexts und die große Hardware-Unterstützung. Die weite Verbreitung, die Linux in den

letzten Jahren gefunden hat, aber auch der Internet-Boom mit dem damit verbundenen großen Bedarf an stabilen und skalierbaren Hochleistungsnetzwerk-Servern, hat der Unix-Welt neuen Auftrieb gegeben.

Hinweis

Unix wird in diesem Buch als Oberbegriff für diverse vom ursprünglichen Unix abgeleitete Betriebssysteme verwendet. Die Namen dieser Betriebssysteme enden im Regelfall auf -ix (Irix, Xenix etc.) und sind zumeist geschützte Warenzeichen der jeweiligen Firmen. UNIX selbst ist ebenfalls ein geschütztes Warenzeichen. Die Rechte haben in den vergangenen Jahren mehrfach den Besitzer gewechselt.

Übrigens können Sie auf Ihrem PC durchaus mehrere Betriebssysteme parallel installieren. Sie müssen dann beim Starten des Rechners angeben, ob der Rechner unter Windows, unter Linux oder unter einem anderen Betriebssystem gstartet werden soll. Unter Linux ist sogar ein Zugriff auf das Windows-Dateisystem möglich.

Der Kernel

Genau genommen bezeichnet der Begriff Linux nur den **Kernel**: Der Kernel ist der innerste Teil (Kern) eines Betriebssystems mit ganz elementaren Funktionen wie Speicherverwaltung, Prozessverwaltung und Steuerung der Hardware. Die Informationen in diesem Buch beziehen sich auf die Kernel 2.4 und 2.6. Bei den Versionsnummern des Kernels wird zwischen stabilen Kernel-Versionen (gerade Ziffer an zweiter Stelle) und Entwickler-Kerneln unterschieden (ungerade Ziffer an zweiter Stelle). Linux-Anwender, die Wert auf ein stabiles System legen, sollten einen ausgereiften und erfahrungsgemäß sehr stabilen Anwender-Kernel verwenden. Nur wer an der Weiterentwicklung des Kernels teilnehmen möchte oder auf ein neues Feature des Entwickler-Kernels angewiesen ist, das im letzten stabilen Kernel noch nicht enthalten ist, muss mit einem Entwickler-Kernel vorlieb nehmen. Die Entwickler-Kernel enthalten oft noch unzureichend getesteten Code; die Wahrscheinlichkeit von Problemen ist daher viel größer. Die meisten Distributionen basieren auf stabilen Kerneln, bieten aber manchmal die Option, auch den jeweils neuesten verfügbaren Entwickler-Kernel zu installieren.

Da Linux ständig weiterentwickelt wird, gibt es praktisch wöchentlich eine oder mehrere neue Versionen des Entwickler-Kernels. Sie brauchen aber keine Angst zu haben, dass Sie ständig irgendwelche Updates nachinstallieren müssen. Sobald Linux einmal stabil auf Ihrem Rechner läuft, besteht selten ein Grund, daran etwas zu ändern. Die meisten Änderungen am Kernel betreffen irgendwelche besonderen Aspekte von Linux, etwa die Anpassung des Systems an neue Hardware, die Korrektur von (zumeist nur unter seltenen Umständen auftretenden) Fehlern oder die Optimierung einer Funktion.

Tipp Generell sollten Sie sich von Versionsnummern nicht allzu sehr verwirren lassen: Nicht nur der Kernel hat eine Versionsnummer, sondern praktisch alle unter Linux verfügbaren Libraries, Compiler, Programme, Distributionen usw.

Kerneleigenschaften

- Linux unterstützt Multitasking (die gleichzeitige Abarbeitung mehrerer Prozesse), Multiuser-Betrieb (die gleichzeitige Nutzung durch mehrere Anwender), Paging (die Auslagerung von Speicher auf die Festplatte, wenn zu wenig RAM zur Verfügung steht), Shared Libraries (Bibliotheken mit Systemfunktionen werden nur einmal geladen, wenn sie von mehreren Prozessen benötigt werden), Interprocess Communication (IPC) und Symmetric Multi Processing (SMP, die Nutzung mehrerer Prozessoren).
- Linux unterstützt praktisch die gesamte gängige PC-Hardware. Neben der Intel-Prozessorplattform gibt es auch Linux-Versionen für zahlreiche andere Rechnerarchitekturen.
- Linux unterstützt eine ganze Menge von Dateisystemen. Am gebräuchlichsten ist `ext3`. So genannte *journaling*-Funktionen stellen sicher, dass das Dateisystem nach einem unvorhergesehenen Rechnerabsturz (etwa nach einem Stromausfall) sehr rasch wieder verwendet werden kann. Selbst in diesem Fall sind Datenverluste ziemlich unwahrscheinlich.
- Unter Linux steht eine ganze Palette von Netzwerkprotokollen zur Verfügung (TCP/IP inklusive IPv6, IPsec, PPP etc.).

Hardware-Unterstützung

Linux unterstützt beinahe die gesamte gängige PC-Hardware. Es gibt aber einige Ausnahmen bzw. typische Problemfälle:

- Sehr neue Grafikkarten werden oft nicht oder nur teilweise unterstüzt. Für manche Grafikkarten stellen die Hardware-Firmen (z. B. ATI, NVIDA) eigene Treiber zur Verfügung, deren Integration in Linux aber mühsam sein kann. Dasselbe trifft auch auf andere Steckkarten zu (z.B. WLAN-Karten).
- Neue Notebooks enthalten oft Spezialkomponenten, die unter Linux nicht oder nur teilweise genutzt werden können. Das betrifft beispielsweise die verschiedenen Energiesparmodi, die über den ACPI-Standard (*Advanced Configuration and Power Interface*) gesteuert werden. Grundfunktionen von Linux laufen meistens, Spezialfunktionen des Notebooks können aber manchmal nur zum Teil genutzt werden.

- Eine weitere Problemquelle ist Windows-spezifische Zusatz-Hardware: GDI-Drucker, WinModems, manche Scanner etc. Diese Geräte unterstützen keine öffentlichen Standards, sondern funktionieren nur mit einem eigens für Windows entwickelten Treiber. Die Entwicklung von Linux-Treibern scheitert oft daran, dass die jeweiligen Hersteller keine Spezifikationen zur Verfügung stellen oder deren Verwendung in freiem Code verbieten.

Aus diesen Gründen sollten Sie *vor* dem Kauf eines neuen Rechners bzw. einer Hardware-Erweiterung sicherstellen, dass alle Komponenten von Linux unterstützt werden. Sie ersparen sich damit viel Ärger! Auf den folgenden Webseiten finden Sie dazu aktuelle Informationen. Ein guter Startpunkt ist das Hardware-HOWTO-Dokument (erster Link).

Überblick:	*http://www.tldp.org/HOWTO/HOWTO-INDEX/hardware.html*
Grafikkarten:	*http://www.XFree86.org*
Drucker:	*http://www.linuxprinting.org*
Scanner:	*http://www.sane-project.org/*
WinModems:	*http://linmodems.org*
WLAN:	*http://www.hpl.hp.com/personal/Jean_Tourrilhes/ Linux/Wireless.html*
USB:	*http://www.linux-usb.org*
Firewire:	*http://www.linux1394.org/*
Notebooks 1:	*http://www.linux-on-laptops.net*
Notebooks 2:	*http://tuxmobil.org/index.html*
Suse:	*http://cdb.Suse.de*
Red Hat:	*http://www.redhat.com/support/hardware*

1.2 Distributionen

Die Antwort „Linux ist der Kernel" mag den Fachmann befriedigen, aber die meisten Anwender interessiert der Kernel nämlich herzlich wenig; er ist halt notwendig. Für sie umfasst der Begriff Linux, wie er umgangssprachlich verwendet wird, neben dem Kernel auch das riesige Bündel von Programmen, das bei einer Linux-Distribution typischerweise mitgeliefert wird: Dazu zählen neben unzähligen kleinen Tools und Werkzeugen die Desktop-Systeme KDE und Gnome, verschiedene Office-Pakete, das Zeichenprogramm Gimp, zahllose Programmiersprachen und mindestens ebenso viele Server-Programme (Webserver, Mail-Server, File-Server, Print-Server etc.).

Als Linux-Distribution wird also die Einheit bezeichnet, die aus dem eigentlichen Betriebssystem (Kernel) und seinen Zusatzprogrammen besteht. Eine Distribution ermöglicht eine rasche und bequeme Installation von Linux. Distributionen werden zumeist in Form von CD-ROMs oder DVDs verkauft. Viele Distributionen sind darüber hinaus auch zum Download im Internet verfügbar. Wegen der

riesigen Datenmengen (oft mehrere GByte) ist das Kopieren einer Distribution via Internet bzw. eine direkte Installation über das Netz aber nur bei einer ausgezeichneten Internet-Anbindung möglich. Manche behandeln die Distributionen sogar so, als seien sie eigene Betriebssysteme (da heisst es dann nicht „läuft unter Linux“, sondern „läuft unter Suse 9.9“.

Distributionen unterscheiden sich vor allem durch folgende Punkte voneinander:

- **Umfang, Aktualität:** Die Anzahl, Auswahl und Aktualität der mitgelieferten Programme und Bibliotheken unterscheidet sich von Distribution zu Distribution. Manche Distributionen überbieten sich in der Anzahl der mitgelieferten CDs oder DVDs.
- **Installations- und Konfigurationswerkzeuge:** Die mitgelieferten Programme zur Installation, Konfiguration und Wartung des Systems helfen dabei, Hunderte von Konfigurationsdateien des Systems auf einfache Weise richtig einzustellen. Funktionierende Werkzeuge für Installation und Konfiguration stellen eine enorme Zeitersparnis dar.
- **Konfiguration des Desktops (KDE, Gnome):** Viele Distributionen lassen dem Anwender die Wahl zwischen verschiedenen grafischen Benutzeroberflächen (Window-Managern). Es gibt aber auch Unterschiede in der Detailkonfiguration von KDE oder Gnome, die das Aussehen, die Menüanordnung etc. betreffen.
- **Hardware-Erkennung und -Konfiguration:** Linux kommt mit den meisten PC-Hardware-Komponenten zurecht. Angenehm ist natürlich, wenn die Distribution Ihre Hardware automatisch erkennt und damit umgehen kann. Gelingt dies nicht, ist oft eine mühsame Konfiguration in Handarbeit erforderlich, die Linux-Einsteiger meist überfordert.
- **Paketsystem:** Die Verwaltung von Linux-Anwendungsprogrammen erfolgt durch Pakete. Das Paketsystem hat Einfluss darauf, wie einfach die Nachinstallation zusätzlicher Programme bzw. das Update vorhandener Programme ist. Zurzeit sind drei zueinander inkompatible Paketsysteme üblich, RPM (unter anderem bei Mandriva, Red Hat, Suse, TurboLinux), DEB (Debian) und TGZ (Slackware).
- **Wartung, Sicherheits-Updates:** Linux ist ein sich dynamisch veränderndes System. Oft gibt es nach der Fertigstellung einer Distribution noch wichtige Neuerungen; immer wieder werden Sicherheitsmängel in diversen mitgelieferten Programmen entdeckt. Eine gute Distribution zeichnet sich dadurch aus, dass es ein halb- oder vollautomatisches Update-System gibt, um Sicherheits-Updates mühelos zu installieren.

  Zunehmend von Bedeutung wird der Zeitraum, während dessen Updates zur Verfügung gestellt werden. Hier gilt meist die Grundregel: je teurer die Distribution, desto länger der Zeitraum. Die Werte liegen zwischen sechs Monaten

und fünf Jahren. Grundsätzlich gilt, dass Sie eine Linux-Distribution nur so lange sicher betreiben können, wie Sie Updates bekommen. Danach ist aus Sicherheitsgründen ein Wechsel auf eine neue Version der Distribution erforderlich.

- **Live-System:** Einige wenige Distributionen ermöglichen den Betrieb von Linux direkt von einer CD-ROM (z.B. Knoppix). Das ist zwar langsam und unflexibel, ermöglicht aber ein vergleichsweise einfaches Ausprobieren von Linux. Zudem stellt eine Live-CD eine ideale Möglichkeit dar, ein auf der Festplatte vorhandenes, aber defektes Linux-System zu reparieren.
- **Zielplattform (CPU-Architektur):** Es gibt auch Distributionen für andere Prozessorplattformen, z. B. für Sparc (SUN) oder für PowerPC (Apple).
- **Dokumentation:** Große Unterschiede gibt es bei der Qualität und dem Umfang der mitgelieferten Handbücher.
- **Support:** Bei manchen Distributionen bekommen Sie kostenlos Hilfe bei der Installation (via E-Mail und/oder per Telefon).
- **Mitgelieferte kommerzielle Software:** Bei manchen Distributionen werden nicht nur die frei verfügbaren Linux-Pakete mitgeliefert, sondern auch lizenzpflichtige Programme.
- **Lizenz:** Die meisten Distributionen sind uneingeschränkt kostenlos über das Internet erhältlich. Bei einigen Distributionen gibt es hier aber Einschränkungen. Beispielsweise stellen nicht alle Distributionen so genannte ISO-Images zur Verfügung, mit denen sich Anwender selbst Installations-CDs brennen können (also ohne die Distribution zu kaufen).

  Manche Distributionen erlauben zwar die kostenlose Weitergabe, nicht aber den Weiterverkauf von CDs. Da Linux und die meisten mitgelieferten Programme an sich frei erhältlich sind, bezieht sich das Verkaufsverbot meist auf die Nutzung von Markenzeichen (Red Hat) oder auf distributionsspezifische Konfigurationsprogramme (YaST bei Suse). Weitergabe-Einschränkungen gibt es auch, wenn mit der Distribution kommerzielle Software mitgeliefert wird.

Die Behauptung, Linux sei frei, steht scheinbar im krassen Widerspruch zu dem Preis, der für die meisten besseren Distributionen verlangt wird. Der Grund ist aber leicht verständlich: Obwohl Linux und die meisten Anwendungsprogramme tatsächlich kostenlos über das Internet bezogen werden können, erfordert die Zusammenstellung einer aktuellen Distribution eine Menge Zeit und Know-how. Ein gutes Installationsprogramm allein (das auch programmiert und gewartet werden muss) ist den Preis einer Distribution oft schon wert! Es kann vor allem Linux-Neulingen eine Menge Zeit bei der Installation und Konfiguration ersparen.

Auch die Produktion von einer oder mehreren CDs, oft begleitet von einem Handbuch, kostet Geld. Nicht zu vernachlässigen ist schließlich das Angebot eines persönlichen Supports bei Installationsproblemen. Teuer wird eine Distribution auch dann, wenn kommerzielle Software mitgeliefert wird.

Die Frage, welche Distribution die beste sei, welche wem zu empfehlen sei etc., artet leicht zu einem Glaubenskrieg aus. Wer sich einmal für eine Distribution entschieden und sich an deren Eigenheiten gewöhnt hat, steigt nicht so schnell auf eine andere Distribution um. Ein Wechsel der Distribution ist nur durch eine Neuinstallation möglich, bereitet also einige Mühe.

Kriterien für die Auswahl einer Distribution sind die Aktualität ihrer Komponenten (achten Sie auf die Versionsnummer des Kernels und wichtiger Programme, etwa des C-Compilers), die Qualität der Installations- und Konfigurations-Tools, der angebotene Support, mitgelieferte Handbücher etc.

Dass viele Linux-Distributionen wirklich uneingeschränkt frei verfügbar sind, erkennen Sie unter anderem daran, dass es selbst von manchen kommerziellen Distributionen Billig-CDs gibt. Diesen Versionen fehlen alle kommerziellen Komponenten, die von der jeweiligen Distribution lizenziert wurden. Außerdem gibt es keine Dokumentation und keinen Support. Lassen Sie sich vom günstigen Preis dieser Versionen nicht blenden. Gerade für Einsteiger ist ein gutes Handbuch oder die Support-Möglichkeit sehr wertvoll.

Einige gängige Linux-Distributionen

Der folgende Überblick über die wichtigsten verfügbaren Distributionen (in alphabetischer Reihenfolge und ohne Anspruch auf Vollständigkeit!) soll eine erste Orientierungshilfe geben. Die Landschaft der Linux-Distributionen ändert sich ständig. Neue Distributionen entstehen, andere verlieren an Bedeutung oder werden ganz eingestellt.

- **Debian:** wird von von engagierten Linux-Anwendern zusammengestellt und vollkommen kostenlos zur Verfügung gestellt. Debian legt größten Wert auf Stabilität und die Einhaltung der Spielregeln „freier“ Software.
- **Fedora** ist das Ergebnis der Reorganisation von Red Hat, deren Ergebnis darin bestand, dass die klassische RedHat-Distribution einerseits durch die ziemlich teuren Enterprise-Versionen und andererseits durch das kostenlose Fedora ersetzt wurde. Die erste Fedora-Version mit dem offiziellen Namen *Fedora Core 1* wurde Anfang November 2003 vorgestellt.
- **Gentoo** richtet sich besonders an Programmentwickler und Anwender, die maximale Flexibilität und Kontrolle über ihre Distribution wünschen. Die Besonderheit von Gentoo besteht darin, dass jedes Programmpaket eigens kompiliert und so optimal an die jeweilige Hardware angepasst werden kann.

- **Knoppix:** Die Besonderheit von Knoppix besteht darin, dass sie nicht auf die Festplatte installiert werden muss. Stattdessen läuft Knoppix direkt von der CD. (Es handelt sich also um ein so genanntes Live-System.) Knoppix ist von Debian abgeleitet.
- **Linspire** (ehemals Lindows) ist eine kommerzielle Distribution mit dem Ziel, einen möglichst hohen Bedienkomfort für Linux-Einsteiger bzw. Windows-Umsteiger zu bieten. In vielen Details lehnen sich das Aussehen und die Bedienung von Linspire an Windows an.
- **Mandriva** (vormals Mandrake) wurde ursprünglich von Red Hat abgeleitet, ist inzwischen aber eine vollkommen eigenständige Distribution, die großen Wert auf Benutzerfreundlichkeit und einfache Bedienung legt.
- **Red Hat** ist die international bekannteste Linux-Firma. RedHat-Distributionen dominieren insbesondere den amerikanischen Markt. Die Paketverwaltung auf der Basis des von Red Hat entwickelten rpm-Formats wurde mittlerweile von vielen anderen Distributionen übernommen. 2003 hat Red Hat seine Distributionen neu organisiert: Auf der einen Seite stehen die relativ teuren Enterprise-Versionen, die sich nur an Firmenkunden richten. (Für Universitäten bzw. Studenten gibt es dieselben Distributionen deutlich günstiger.) Auf der anderen Seite steht das kostenlose Fedora-Linux (siehe oben). Neue Enterprise-Versionen sollen für Linux-Verhältnisse nur relativ selten erscheinen (etwa alle 18 Monate), dafür aber sehr lange durch Updates unterstützt werden (fünf Jahre).
- **Slackware**: Slackware war eine der ersten verfügbaren Linux-Distributionen. Bezüglich Wartung und Installationskomfort kann sie allerdings nicht mehr mit den anderen hier genannten Distributionen mithalten.
- **Suse/Novell:** Suse-Linux ist dank der hohen Aktualität, der riesigen Anzahl vorkonfigurierter Pakete, der umfassenden Handbücher und der hervorragenden Wartung die in Europa und insbesondere in Deutschland am weitesten verbreitete Distribution. Das Administrationsprogramm YaST hilft nicht nur bei vielen Konfigurationsproblemen, es löst auch ähnlich wie Debian automatisch Abhängigkeitskonflikte, die bei der Paketinstallation auftreten können. YaST untersteht inzwischen der GPL. Aus diesem Grund darf Suse-Linux nicht nur im Freundeskreis frei kopiert werden, es ist auch zulässig, Suse-CDs billig zu verkaufen.
- **Ubuntu:** Es gab in der Vergangenheit zahlreiche Versuche, neue und benutzerfreundlichere Distributionen von Debian abzuleiten. (Knoppix, Linspire und Xandros sind nur einige Beispiele dafür.) Inzwischen gibt es eine weitere Debian-Variante mit dem Namen „Ubuntu“, deren Schwerpunkt im komfortablen Endbenutzer-Desktop liegt – eine kostengünstige Alternative zu den teilweise recht teuren kommerziellen Linux-Desktop-Distributionen.

- **Xandros** ist eine kommerzielle Distribution, die aus Corel-Linux entstanden ist. Xandros ist im Hinblick auf möglichst einfache Bedienung optimiert und richtet sich an Linux-Einsteiger und Windows-Umsteiger. In der Deluxe-Version enthält Xandros das Produkt „Crossover Office“, das die Installation und den Betrieb von Microsoft Office unter Linux ermöglicht.

Hinweis

Einen ziemlich guten Überblick über die momentan verfügbaren Linux-Distributionen (egal ob kommerziellen oder anderen Ursprungs) finden Sie im Internet auf den folgenden Seiten:

http://www.linuxhq.com/dist-index.html
http://www.distrowatch.com/
http://www.lwn.net

Neben diesen großen Distributionen gibt es im Internet einige Zusammenstellungen von Miniatursystemen (bis hin zum kompletten Linux-System auf einer einzigen Diskette!). Diese Distributionen basieren zumeist auf einem alten und daher kleineren Kernel. Sie sind vor allem für Spezialaufgaben konzipiert, etwa für Wartungsarbeiten (Emergency-System) oder um ein Linux-System ohne eigentliche Installation verwenden zu können (direkt von einer oder mehreren Disketten).

Die Qual der Wahl

Es ist schwierig, hier eine Empfehlung zu geben. Gerade für Linux-Einsteiger ist es aber zumeist ein Vorteil, sich vorerst für eine der weit verbreiteten Distributionen wie Suse, Mandriva oder Red Hat zu entscheiden, weil es bei Problemen vergleichsweise leichter ist, Hilfe zu finden. Diesem Buch liegt eine Debian-Distribution bei, auf die wir in den kommenden Kapiteln noch genauer eingehen werden.

1.3 Traum und Wirklichkeit

Dieser Abschnitt nimmt zu einigen oft gehörten Behauptungen und Vorurteilen zu Linux Stellung. Unser Ziel ist es, Ihnen ein abgerundetes Bild von Linux zu präsentieren, ohne die Übertreibungen vieler begeisterter Linux-Fans, aber auch ohne die Schwarzmalerei der Linux-Gegner.

Linux ist schneller/langsamer als Windows: Diese Aussage ist so weder in der einen noch in der anderen Form richtig. Tatsächlich gibt es einzelne Programme, die unter Linux oder unter Windows schneller laufen. Daraus lassen sich aber keine allgemein gültigen Schlussfolgerungen ziehen. Das Ergebnis

hängt unter anderem davon ab, für welches Betriebssystem das Programm optimiert wurde, welche Linux- und Windows-Versionen miteinander verglichen werden, welche Hardware für den Vergleich verwendet wurde etc.

Linux benötigt weniger Ressourcen als Windows: Grundsätzlich stimmt es, dass Sie Linux auf einem 486-er PC mit einigen MByte RAM betreiben können. In dieser Konfiguration läuft Linux zwar nur im Textmodus, bietet ansonsten aber sicher viel mehr Funktionen als eine alte Windows-Version, die auf einem derartigen Rechner ebenfalls noch läuft.

Wenn Sie dagegen eine aktuelle Linux-Distribution von Red Hat oder Suse mit einer aktuellen Windows-Version vergleichen, sind die Unterschiede weniger deutlich. Für ein komfortables Arbeiten in einer grafischen Benutzeroberfläche (KDE oder Gnome) stellt Linux ähnliche Hardware-Ansprüche wie Windows.

Linux ist sicherer als Windows: Leider kranken alle zurzeit populären Betriebssysteme an Sicherheitsproblemen. Linux schneidet in den meisten Vergleichen relativ gut ab, dennoch finden sich immer wieder neue Sicherheitslücken. Wie sicher Linux ist, hängt aber auch von seiner Verwendung ab:

- In Desktop-Anwendungen ist Linux fast vollständig virensicher. Es hat bis jetzt keinen einzigen nennenswerten Virenbefall unter Linux gegeben, während Windows-Viren in regelmäßigen Abständen ganze Firmen tagelang lähmen. Der Hauptgrund besteht darin, dass die Zugriffsverwaltung unter Linux verhindert, dass gewöhnliche Anwender großen Schaden am System anrichten können. Außerdem sind Webbrowser und E-Mail-Programme unter Linux generell viel sicherer als die entsprechenden Windows-Programme.
- Bei der Anwendung von Linux als Netzwerk- oder Internet-Server hängt die Sicherheit sehr stark von der Wartung des Systems ab. Beinahe zu allen Sicherheitsproblemen der vergangenen Jahre gab es bereits Updates, bevor diese Sicherheitsrisiken allgemein bekannt und von Hackern ausgenutzt wurden. Wenn Sie also die auf Ihrem Rechner eingesetzte Software regelmäßig aktualisieren, haben Angreifer wenig Chancen, in Ihr System einzudringen.

Die Sicherheit von Linux-Systemen hängt schließlich sehr stark von Ihrem eigenen Wissen ab. Wenn Sie als Linux-Einsteiger rasch einen Internet-Server konfigurieren und ins Netz stellen, ist nicht zu erwarten, dass dieser Server bereits optimal abgesichert ist. Es mangelt aber nicht an Literatur zu diesem Thema!

Linux ist stabiler als Windows: Mittlerweile hat Microsoft mit Windows 2000/XP durchaus respektable und stabile Windows-Versionen zustande gebracht. Der Linux-Kernel an sich ist außerordentlich stabil. Wenn Sie mit Linux aber das Gesamtsystem der mitgelieferten Software meinen (also eine ganze Distribution), dann sieht es mit der Stabilität gleich erheblich schlechter aus. Insbesondere relativ neue Programme stürzen immer wieder ab. Server-Programme laufen dagegen meist vollkommen fehlerfrei. Je stärker Sie sich anwendungsorientierten Programmen zuwenden und Linux als Desktop-System einsetzen, desto eher werden Sie die negativen Seiten kennen lernen.

Linux ist billiger als Windows: Diese Aussage ist leicht zu untermauern – Linux ist schließlich kostenlos erhältlich. Bei Microsoft hat man mit dieser Argumentation natürlich keine Freude – dort weist man darauf hin, dass auch Schulungskosten etc. berücksichtigt werden müssen. (In solchen Rechenbeispielen wird Windows-Wissen meist als gottgegeben vorausgesetzt, Linux-Kenntnisse natürlich nicht.) Außerdem ist nicht jede Linux-Distribution tatsächlich kostenlos.

Linux ist kompliziert zu installieren: Wenn man einen PC kauft, ist Windows meist schon vorinstalliert. Insofern stellt es natürlich einen Mehraufwand dar, Linux zusätzlich zu installieren. Wie Sie im nächsten Kapitel feststellen werden, ist eine Linux-Installation aber mittlerweile kinderleicht – und sicher nicht schwieriger als eine Windows-Installation. Problematisch ist lediglich die Unterstützung neuer Hardware, die unter Windows besser ist: Jeder Hersteller von Computer-Komponenten stellt selbstverständlich einen Windows-Treiber zur Verfügung. Vergleichbare Treiber für Linux müssen dagegen oft von der Open-Source-Gemeinschaft programmiert werden. Das dauert natürlich eine gewisse Zeit.

Linux ist kompliziert zu bedienen: Dieses Vorurteil ist alt, aber nicht mehr bzw. nur noch in einem sehr geringen Maß zutreffend. Linux ist einfach anders zu bedienen als Windows, so wie auch Apples Mac OS anders zu bedienen ist. Wirklich schwieriger ist die Handhabung von Linux zumeist nicht, lediglich die Umgewöhnung von Windows kann manchmal mühsam sein.

Linux ist ungeeignet für Multimedia-Anwendungen: Ob diese Aussage zutrifft, hängt stark davon ab, was Sie unter „Multimedia" verstehen: Auf der einen Seite hat Linux keine Probleme beim Abspielen von Audio-Dateien (MP3). Es kann selbst Audio-Dateien kodieren (allerdings per Default nur im Ogg-Vorbis-Format), kann als digitaler Video-Rekorder konfiguriert werden, kann CDs und DVDs brennen etc. Die Verwaltung, Bearbeitung und der Ausdruck digitaler Bilder gelingt problemlos.

Auf der anderen Seite gibt es für Linux momentan kein DRM-System (*Digital Rights Management*), das vermutlich in naher Zukunft für jeden legalen Audio- und Video-Download notwendig sein wird. Die meisten Linux-Distributionen liefern aus rechtlichen Gründen auch keinen DVD-Player mehr mit (Entsprechende Programme gibt es aber im Internet). Es gibt weniger Spiele für Linux. Zudem ist die Nutzung der 3D-Funktionen moderner Grafikkarten relativ schwierig. Auch die Verwendung anderer moderner Multimedia-Hardware bereitet unter Linux häufig Probleme.

Windows-Programme können unter Linux nicht ausgeführt werden: Diese Aussage ist grundsätzlich richtig. Viele Programme wie Microsoft Office, Adobe Photoshop etc. stehen momentan nur für die Betriebssysteme Windows und Mac OS zur Verfügung. Es gibt aber einige Auswege aus diesem Software-Dilemma:

- Für viele Anwendungen stehen unter Linux vergleichbare Programme zur Verfügung – beispielsweise OpenOffice oder das Bildverarbeitungsprogramm Gimp. Diesem Themenkomplex ist ein eigenes Kapitel gewidmet.
- Manche (zumeist nur recht einfache) Windows-Programme können mit dem kostenlosen Windows-Emulator Wine unter Linux ausgeführt werden.
- Einen höheren Grad an Kompatibilität bietet das kommerzielle Programm CrossOver Office, das auf Wine basiert. CrossOver Office ermöglicht die Installation und Ausführung der meisten Microsoft-Office-Komponenten sowie einiger anderer Programme direkt unter Linux.
- Das Programm VMWare geht noch einen Schritt weiter: Es emuliert einen ganzen Rechner. Sie können darin eine Windows-Installation durchführen und Windows dann in einem Fenster ausführen. Das funktioniert an sich hervorragend, erfordert aber eine gute Hardware (insbesondere viel RAM) und ist relativ teuer: Zum Preis für VMWare kommt noch die Lizenz für Windows hinzu.

1.4 Herkunft und Lizenz von Linux

In diesem Abschnitt geht es weniger um die Details von Linux als um die Fragen, warum es Linux überhaupt gibt, wie das System entwickelt wurde und warum es frei ist. Dabei spielen GNU (*GNU is Not Unix*) und die FSF (*Free Software Foundation*) eine besondere Rolle.

Die allerersten Teile des Linux-Kernels (Version 0.01) wurden von Linus Torvalds (Helsinki) entwickelt, der den Programmcode im September 1991 über das Internet freigab. In kürzester Zeit fanden sich weltweit Programmierer, die an der Idee Interesse hatten und Erweiterungen dazu programmierten: ein verbessertes System zur Dateiverwaltung, Treiber für diverse Hardware-Komponenten, Zusatzprogramme wie den DOS-Emulator etc. All diese Einzelkomponenten wurden ebenfalls kostenlos zur Verfügung gestellt. Das Gesamtsystem wuchs mit einer atemberaubenden Geschwindigkeit. Die Entstehung dieses neuen Betriebssystems wäre ohne die weltweite Kommunikation der Programmierer via Internet unmöglich gewesen.

Ein wesentlicher Faktor dafür, dass Linux frei von den Rechten der großen Software-Firmen ist und dennoch derart schnell entwickelt werden konnte, war die zu diesem Zeitpunkt schon frei verfügbare Software. Linux ist nicht aus dem Nichts aufgetaucht, wie das manchmal fälschlich dargestellt wird, sondern baut auf einer breiten Basis freier Software auf. Für die ersten Schritte war das freie (aber im Funktionsumfang sehr eingeschränkte) Minix eine praktische Grundlage. So verwendeten die ersten Linux-Versionen noch das Dateisystem von Minix.

In ihrer Bedeutung wohl noch wichtiger waren und sind die zahlreichen GNU-Programme. GNU-Programme gibt es schon wesentlich länger als Linux. Schon

vorher wurden GNU-Programme auf vielen Unix-Systemen als Ersatz für diverse Originalkomponenten verwendet – etwa der GNU-C-Compiler, der Texteditor Emacs, diverse GNU-Utilities wie `find` und `grep` etc. Sobald der Kernel von Linux so weit entwickelt worden war, dass der GNU-C-Compiler darauf zum Laufen gebracht werden konnte, stand praktisch mit einem Schlag die gesamte Palette der GNU-Tools zur Verfügung. Aus dem bloßen Kernel wurde also plötzlich ein recht vollständiges System, das dann für eine noch größere Entwicklergemeinde zu einer attraktiven Umgebung wurde.

GNU-Programme sind ebenso wie Linux (unter gewissen Einschränkungen) frei kopierbar – und zwar nicht nur als Binärprogramme, sondern mit sämtlichen Codequellen. Das ermöglicht es allen GNU-Anwendern, die Programme bei Problemen oder Fehlern selbst zu erweitern oder zu korrigieren. Aus diesen Änderungen resultieren immer bessere und ausgereiftere Versionen der diversen GNU-Programme. Nicht zuletzt aufgrund der freien Verfügbarkeit des Programmcodes stellt der GNU-C-Compiler *den* Standard in der Unix-Welt dar: Der Compiler ist praktisch auf jedem Unix-System verfügbar. (Es gibt übrigens auch Portierungen für Windows.) Linux selbst ist mit dem GNU-C-Compiler entstanden.

Erst die Kombination aus dem Linux-Kernel, den zahlreichen GNU-Komponenten, der Netzwerk-Software des BSD-Unix, dem ebenfalls frei verfügbaren X Window System des MIT (Massachusetts Institute of Technology) und dessen Portierung XFree86 für PCs mit Intel-Prozessoren sowie aus zahlreichen weiteren Programmen macht eine Linux-Distribution zu einem kompletten Unix-System.

Linux ist natürlich nicht allein Linus Torvalds zu verdanken (auch wenn es ohne ihn Linux in seiner heutigen Form nicht gäbe). Hinter Linux stehen vielmehr eine Menge engagierter Menschen, die seit Jahren in ihrer Freizeit (seltener auch im Rahmen ihres Informatikstudiums) freie Software produzieren. (Der Kernel von Linux umfasst mittlerweile mehrere Millionen Programmzeilen!) Die wichtigsten Namen sind in der Datei `/usr/src/linux/CREDITS` genannt. Diese Datei können Sie nach der Installation des Linux-Kernels lesen.

Rechtliches – Die General Public License

Linux ist frei – aber was bedeutet das eigentlich? Oft wird „frei" mit „kostenlos" verwechselt. Der Begriff „frei" bezieht sich aber auch und vor allem auf die Verfügbarkeit des gesamten Quellcodes (*open source*). Damit sind gewisse Komplikationen verbunden: Was passiert, wenn eine Firma den Linux-Code verwendet, in einigen Punkten erweitert und das System anschließend verkauft? Auch das ist erlaubt, allerdings mit einer Einschränkung: Der Programmcode des neuen Systems muss abermals frei verfügbar sein. Diese Regelung stellt sicher, dass Erweiterungen am System allen Anwendern zugute kommen.

Das Ziel der Entwickler von GNU und Linux war es also, ein System zu schaffen, dessen Quellen frei verfügbar sind und es auch bleiben. Um einen Missbrauch auszuschließen, ist Software, die im Sinne von GNU entwickelt wurde und wird,

durch die *GNU General Public License* (kurz GPL) geschützt. Hinter der GPL steht die *Free Software Foundation* (FSF). Diese Organisation wurde von Richard Stallmann (der unter anderem auch Autor des Editors Emacs ist) gegründet, um qualitativ hochwertige Software frei verfügbar zu machen.

Die Kernaussage der GPL besteht darin, dass zwar jeder den Code verändern und sogar die resultierenden Programme verkaufen darf, dass aber gleichzeitig der Anwender/Käufer das Recht auf den vollständigen Code hat und diesen ebenfalls verändern und wieder kostenlos weitergeben darf. Jedes GNU-Programm muss zusammen mit der vollständigen GPL weitergegeben werden. Die GPL-geschützte Software ist also nicht mit Public-Domain-Software zu verwechseln, die vollkommen ungeschützt ist.

Die GPL schließt damit aus, dass jemand ein GPL-Programm weiterentwickeln und verkaufen kann, *ohne* die Veränderungen öffentlich verfügbar zu machen. Jede Weiterentwicklung ist somit ein Gewinn für *alle* Anwender.

Hinweis

Den vollständigen Text der GPL finden Sie unter
http://www.gnu.org/licenses/gpl.html

Die Grundidee der GPL ist recht einfach zu verstehen, im Detail treten aber immer wieder Fragen auf. Viele dieser Fragen werden hier beantwortet:

http://www.gnu.org/licenses/gpl-faq.html

Wenn Sie glauben, dass Sie alles verstanden haben, sollten Sie das GPL-Quiz ausprobieren:

http://www.gnu.org/cgi-bin/license-quiz.cgi

Neben der GPL existiert noch die Variante LGPL (*GNU Lesser GPL*). Der wesentliche Unterschied zur GPL besteht darin, dass eine derart geschützte Bibliothek auch von kommerziellen Produkten genutzt werden darf, deren Code *nicht* frei verfügbar ist. Ohne die LGPL könnten GPL-Bibliotheken nur wieder für GPL-Programme genutzt werden, was in vielen Fällen eine unerwünschte Einschränkung für kommerzielle Programmierer wäre.

Seit Anfang 2006 wird eine Aktualisierung der GPL diskutiert, die bessere an die Problematik der Softwarepatente und andere aktuelle Probleme angepasst sein wird.

Andere Lizenzen

Durchaus nicht alle Teile einer Linux-Distribution unterliegen den gleichen Copyright-Bedingungen! Obwohl der Kernel und viele Tools der GPL unterliegen, gelten für manche Komponenten und Programme andere rechtliche Bedingungen:

- Beispielsweise gibt es für das X Window System eine eigene Lizenz. (Das X Window System wurde ursprünglich von der amerikanischen Universität MIT entwickelt. Die jetzige Lizenz ist von einer früheren Lizenz des MIT abgeleitet.)
- Für manche Netzwerk-Tools gilt die BSD-Lizenz. BSD ist wie Linux ein freies Unix-System. Die BSD-Lizenz ist insofern liberaler als die GPL, als die kommerzielle Nutzung ohne die Freigabe des Codes zulässig ist. Die Lizenz ist daher vor allem für kommerzielle Programmierer interessant, die Produkte entwickeln möchten, deren Code sie nicht veröffentlichen müssen.
- Für einige Programme gelten Doppellizenzen. Beispielsweise können Sie den Datenbank-Server MySQL für Open-Source-Projekte bzw. für die innerbetriebliche Anwendung gemäß der GPL kostenlos einsetzen. Wenn Sie hingegen ein kommerzielles Produkt auf der Basis von MySQL entwickeln und samt MySQL verkaufen möchten (ohne Ihren Quellcode zur Verfügung zu stellen), kommt die ebenfalls verfügbare kommerzielle Lizenz zum Einsatz. (Das bedeutet, dass die Weitergabe von MySQL in diesem Fall kostenpflichtig wird.)
- Einige wenige unter Linux verfügbare Programme sind Shareware (etwa das beliebte Grafikprogramm `xv`): Die private Nutzung von `xv` ist kostenlos, aber die kommerzielle Anwendung des Programms ist nur nach einer Registrierung beim Autor gestattet.
- Andere Programme sind zwar kommerziell, es ist aber dennoch eine kostenlose Nutzung möglich. Ein bekanntes Beispiel ist der Adobe Reader zum Lesen von PDF-Dokumenten: Zwar ist das Programm unter Linux kostenlos erhältlich (und darf auch in Firmen kostenlos eingesetzt werden), aber der Quellcode zu diesem Programm ist nicht verfügbar.

Manche Distributionen kennzeichnen die Produkte, bei denen die Nutzung oder Weitergabe eventuell lizenzrechtliche Probleme verursachen könnte. Bei Debian befinden sich solche Programme im Verzeichnis `non-free`.

Im Allgemeinen können Sie davon ausgehen, dass Sie alle Programme, die Sie mit einer Linux-Distribution erhalten haben, auch kostenlos nutzen dürfen. Es ist aber nicht immer so, dass Sie davon abgeleitete eigene Produkte ohne weiteres weiterverkaufen dürfen. Wenn Sie Software-Entwickler sind, müssen Sie sich in die bisweilen sehr verwirrende Problematik der unterschiedlichen Software-Lizenzen einarbeiten.

Der Begriff Open Source

Das Dickicht der zahllosen, mehr oder weniger „freien“ Lizenzen ist schwer zu durchschauen. Die Bandbreite zwischen der manchmal fundamentalistischen Auslegung von „frei“ im Sinne der GPL und den verklausulierten Bestimmungen mancher Firmen, die ihr Software-Produkt zwar frei nennen möchten (weil dies gerade modern ist), in Wirklichkeit aber uneingeschränkte Kontrolle über den Code behalten möchten, ist groß.

Eine gute Einführung in das Thema gibt die folgende Website. Das Ziel der Website ist es, unabhängig von Einzel- oder Firmeninteressen die Idee (oder das Ideal) von Software mit frei verfügbarem Quellcode zu fördern. Dort finden Sie auch eine Liste von Lizenzen, die der Open-Source-Idee entsprechen.

http://www.opensource.org

Kapitel 2

Installation

Einführung

Dieses Kapitel gibt einen Überblick über die Installation eines Linux-Systems auf einem PC mit Intel-kompatiblem Prozessor. Das Kapitel bezieht sich größtenteils nicht auf eine spezielle Distribution, sondern beschreibt die wesentlichen Installationsschritte in allgemeiner Form und vermittelt das erforderliche Grundlagenwissen (damit Sie wissen, was Sie tun).

Die Installation von Linux ist in den vergangenen Jahren immer einfacher geworden. Im Idealfall – d. h. wenn Sie Standard-Hardware verwenden und bereits ausreichend Platz für Linux vorhanden ist – sollten 30 Minuten ausreichen, um zu einem funktionierenden Linux-System zu gelangen. Schwierig wird die Installation zumeist nur deswegen, weil im Regelfall ein wechselweiser Betrieb von einem schon vorhandenen Windows-Betriebssystem und von Linux gewährleistet werden soll. Probleme kann es aber auch bei der Unterstützung ungewöhnlicher Hardware geben (z. B. bei sehr alter oder sehr neuer Hardware oder bei manchen Notebooks).

Anmerkungen zur Installation von Linux mit den beiliegenden DVDs finden Sie im Anhang.

2.1 Voraussetzungen

Damit Sie Linux installieren können, müssen mehrere Voraussetzungen erfüllt sein:

- Sie benötigen einen PC mit einem Intel-Pentium-kompatiblen Prozessor. (Es gibt auch Linux-Distributionen für Systeme mit anderen Prozessoren; diese werden hier aber nicht behandelt.)

- Sie benötigen genug RAM. Wenn Sie unter Linux das Desktop-System KDE oder Gnome benutzen möchten, sind 256 MByte ein guter Richtwert.
- Sie benötigen eine freie Partition mit ausreichend Platz auf Ihrer Festplatte. Wie viel „ausreichend“ ist, hängt natürlich von der Distribution und davon ab, wie viele Programme Sie installieren. Ein bis zwei GByte sollten es auf jeden Fall sein, ca. fünf GByte oder mehr wären besser.
- Sie benötigen Hardware-Komponenten, die von Linux erkannt und unterstützt werden. Gegenwärtig ist das bei einem Großteil der Standard-Hardware der Fall. Probleme können ganz neue Grafikkarten, WLAN-Karten oder Notebooks bereiten. Nicht unterstützt werden GDI-Drucker und Win-Modems.
- Manche X-Window-Programme nutzen die mittlere Maustaste. Insofern ist eine Maus mit drei Tasten oder eine Maus mit einem drückbaren Rad praktisch. (Bei Mäusen mit nur zwei Tasten müssen beide Tasten gemeinsam gedrückt werden, um die fehlende dritte Taste zu simulieren.)

2.2 Überblick über den Installationsprozess

Dieser Abschnitt fasst die Schritte einer gewöhnlichen Linux-Installation zusammen. Gewöhnlich bedeutet hier, dass auf dem Rechner bereits Microsoft Windows installiert ist und dass Sie als Installationsmedium eine CD-ROM oder DVD verwenden. (Die Installation ist übrigens wesentlich einfacher, wenn auf dem Rechner noch kein Betriebssystem installiert ist oder wenn dieses gelöscht werden darf.)

- **Windows-Partition verkleinern:** Normalerweise füllt Windows die gesamte Festplatte in einer einzigen, sehr großen Partition. Um Platz für Linux zu machen, muss diese Partition verkleinert werden. Bei einigen Distributionen kümmert sich das Installationsprogramm um diesen Schritt. Andernfalls müssen Sie selbst Hand anlegen (siehe Seite 47). Besser ist es, für Linux eine zweite Platte vorzusehen.
- **Linux-Installation starten:** Legen Sie die Linux-Installations-CD oder -DVD in das Laufwerk ein, und starten Sie den Rechner neu. Das Linux-Installationsprogramm sollte automatisch gestartet werden (siehe Seite 51). Gegebenenfalls muss im BIOS-Setup die Boot-Reihenfolge umgestellt werden. Das Installationsprogramm sieht bei jeder Distribution ein wenig anders aus. Die ersten Fragen betreffen zumeist die Sprache der Benutzeroberfläche sowie die Konfiguration von Tastatur und Maus.
- **Linux-Partitionen anlegen:** Ein wesentlicher Schritt einer jeden Installation – unabhängig von der Distribution, die Sie verwenden – ist das Anlegen von Linux-Partitionen auf der Festplatte. Wie das Partitionierprogramm

aussieht, hängt stark von der jeweiligen Distribution ab. Davon unabhängig finden Sie auf Seite 53 Grundlageninformationen und Tipps zur Partitionierung.

- **Installationsumfang auswählen:** Im nächsten Schritt geben Sie an, welche Teile der Linux-Distribution Sie installieren möchten (siehe Seite 55). Die marktüblichen Linux-Distributionen sind so umfangreich, dass es nicht sinnvoll ist, einfach alles zu installieren.

- **Konfiguration:** Je nach Installationsprogramm folgen nun diverse Rückfragen zur Konfiguration – z. B. zum gewünschten Passwort für den Administrator `root`, zu den Netzwerkeinstellungen, zur Druckerkonfiguration etc. (siehe Seite 56).

- **Boot-Loader:** Ungeklärt ist jetzt nur noch eine Frage: Wie soll Linux in Zukunft gestartet werden? Dazu wird bei den meisten Distributionen das Programm GRUB eingesetzt (siehe Seite 58). GRUB kann wahlweise in der Boot-Partition der Festplatte, in der Linux-Partition der Festplatte oder auf eine Diskette installiert werden. Am komfortabelsten ist die erste Variante: In diesem Fall erscheint der Boot-Loader bei jedem Start und Sie können auswählen, welches Betriebssystem Sie starten möchten. Der Nachteil besteht allerdings darin, dass die GRUB-Installation in seltenen Fällen Konflikte mit dem bisher installierten Boot-Loader verursachen kann.

  Falls Sie noch einen Rechner mit Diskettenlaufwerk besitzen und Ihre Distribution diese Variante vorsieht, ist die Boot-Loader-Installation auf eine leere Diskette am sichersten. Wenn Sie einmal ein laufendes Linux-System besitzen und ein wenig Erfahrung gewonnen haben, können Sie GRUB später immer noch auf der Festplatte installieren.

Insgesamt wird die Erstinstallation von Linux vermutlich etwa eine Stunde in Anspruch nehmen. Anschließend können Sie mit Linux zu arbeiten beginnen bzw. manuell weitere Konfigurationsschritte durchführen und Linux optimal an Ihre besonderen Ansprüche anpassen.

Tipp

Es gibt während einer Linux-Installation nur zwei kritische Phasen, in denen Sie unbeabsichtigt Daten anderer Betriebssysteme zerstören oder Ihren Rechner nicht mehr startbar machen können: bei der Partitionierung der Festplatte und bei der Installation des Boot-Loaders auf der Festplatte. Führen Sie diese Schritte also mit besonderer Vorsicht aus.

Besonders für Linux-Einsteiger ist es zu empfehlen, dass Sie während der Installation durchgeführte Eingaben protokollieren. Besonders wichtig sind Partitionsnamen (z. B. `/dev/hda5`), die Sie unter Umständen später wieder brauchen.

2.3 Grundlagen der Festplattenpartitionierung

2.3.1 Was sind Partitionen und wozu dienen sie?

Partitionen sind Abschnitte auf der Festplatte. Sie können logisch wie einzelne Festplatten behandelt werden. Windows-Partitionen bekommen eigene Laufwerks-Buchstaben (`C:`, `D:` etc.) und verhalten sich dann scheinbar wirklich wie selbstständige Festplatten.

Im einfachsten Fall gibt es nur eine einzige Partition, die einfach die gesamte Festplatte umfasst. Wenn Sie einen vorkonfigurierten PC mit einem Windows-Betriebssystem kaufen, ist das der Regelfall. Mehrere Partitionen benötigen Sie, sobald Sie mehrere Betriebssysteme gleichzeitig auf Ihrem Rechner installieren möchten. Dabei muss es sich durchaus nicht um Linux handeln! Auch wenn Sie verschiedene Windows-Varianten gleichzeitig installieren möchten, benötigen Sie zwei Partitionen. Der Grund dafür besteht darin, dass unterschiedliche Betriebssysteme im Regelfall auch unterschiedliche Dateisysteme verwenden. Selbst wenn mehrere Betriebssysteme das gleiche Dateisystem unterstützen, ist es zumeist unumgänglich, die Betriebssysteme in unterschiedlichen Partitionen zu installieren, um Doppelgleisigkeiten bei den Verzeichnisnamen zu vermeiden.

Unter Linux kommt noch hinzu, dass es zumeist sinnvoll ist, für Linux selbst mehrere Partitionen vorzusehen – z. B. eine Partition für das Betriebssystem, eine für die eigenen Daten und eine als so genannte Swap-Partition. Dabei handelt es sich um das Gegenstück zur Auslagerungsdatei von Windows.

Um es nochmals zu verdeutlichen: Für eine Linux-Installation kommt es nicht darauf an, wie viel Platz auf Ihrer Festplatte unter Windows noch frei ist. Diesen Platz – innerhalb einer Windows-Partition – können Sie nämlich für Linux nicht nutzen. Sie benötigen für die Linux-Installation Platz außerhalb der Windows-Partition, um dort neue Partitionen anzulegen.

2.3.2 Partitionstypen

Es gibt drei Typen von Festplattenpartitionen: primäre, erweiterte und logische Partitionen.

Auf einer Festplatte können maximal vier primäre Partitionen existieren. Außerdem besteht die Möglichkeit, statt einer dieser vier primären Partitionen eine erweiterte Partition zu definieren. Innerhalb der erweiterten Partition können dann mehrere logische Partitionen angelegt werden. (Der Sinn von erweiterten und logischen Partitionen besteht nur darin, das historisch vorgegebene Limit von nur vier primären Partitionen zu umgehen.)

Eine erweiterte Partition dient nur als Container für logische Partitionen. Zur eigentlichen Speicherung von Daten sind nur primäre und logische Partitionen geeignet.

Die maximale Gesamtanzahl der Partitionen hängt unter Linux vom Typ der Festplatte ab: Bei IDE-Festplatten sind insgesamt bis zu 64 Partitionen möglich, davon maximal 60 logische. Bei SCSI-Festplatten sind dagegen nur 16 Partitionen zulässig, davon maximal 12 logische.

2.3.3 Partitionierung und Formatierung

Um die Aufteilung der Festplatte zu verändern, sieht jedes Betriebssystem eigene Werkzeuge vor. Unter Windows 9x/ME ist es noch immer das aus DOS-Zeiten bekannte Programm `FDISK`. Unter Windows NT/2000/XP steht ein komfortableres grafisches Werkzeug zur Verfügung; der Aufruf ist allerdings bei einer jeden dieser Windows-Versionen ein wenig anders. Unter Linux wird zur Partitionierung entweder `fdisk` oder ein äquivalentes Werkzeug mit grafischer Benutzeroberfläche verwendet.

Mit jeder primären oder logischen Partition wird eine Zusatzinformation (eine Kennzahl) gespeichert, die angibt, welches Dateisystem sich in der Partition befindet. Anhand dieser Information können Betriebssysteme (Windows, Linux etc.) sehr leicht erkennen, welche Partitionen wie angesprochen werden sollen. Partitionen mit fremden Betriebssystemen werden im Regelfall einfach ignoriert.

Durch das *Partitionieren* wird allerdings noch kein Dateisystem erzeugt. Nach der Partitionierung ist der Platz auf der Festplatte also nur reserviert, kann aber noch nicht unmittelbar verwendet werden. Dazu muss die Partition zuerst *formatiert* werden (unter Windows über ein Kontextmenü im Explorer oder mit dem Programm `FORMAT`, unter Linux mit dem Programm `mke2fs` oder seinen „Verwandten“).

Linux unterstützt verschiedene eigene Dateisysteme. Am gebräuchlichsten ist das Dateisystem *ext3*, das mit dem oben erwähnten Programm `mke2fs` angelegt wird. Eine beliebte Alternative ist das *reiserfs*-Dateisystem. Daneben gibt es auch andere Linux-Dateisysteme, die für Spezialaufgaben gedacht sind.

Hinweis

Im Regelfall gehen sowohl durch die Partitionierung als auch durch das Formatieren alle in der betroffenen Partition gespeicherten Daten verloren! Die einzige Ausnahme sind spezielle Werkzeuge zur verlustfreien Verkleinerung von Partitionen (siehe ab Seite 47).

2.3.4 Partitionsnamen unter Windows

Unter Windows werden Partitionen, die das Betriebssystem nutzen kann, mit Laufwerksbuchstaben bezeichnet. `A:` und `B:` sind für Disketten reserviert. Die weiteren Buchstaben bezeichnen die primären und logischen Partitionen der Festplatte. (Erweiterte Partitionen erhalten keinen Laufwerksbuchstaben und sind somit unsichtbar.)

Verwirrend wird die Benennung der Partitionen, wenn es mehrere Festplatten oder andere Laufwerke (CD-ROM, DVD etc.) gibt. In diesem Fall bekommen zuerst die primären Partitionen aller Festplatten bzw. Laufwerke einen Buchstaben. Erst anschließend werden auch die logischen Partitionen benannt. Wenn Sie drei Festplatten mit je einer primären und zwei logischen Windows-Partitionen besitzen, dann gehören also C, F und G zur ersten Platte, D, H und I zur zweiten Platte und E, J und K zur dritten Platte.

Partitionen mit fremden Dateisystemen bekommen keinen Laufwerksbuchstaben und sind daher in den meisten Programmen (z. B. im Explorer) unsichtbar. Die Partitionen werden nur in Partitionierungsprogrammen angezeigt (FDISK unter Windows 9x/ME, Computer-Management-Console unter Windows 2000 etc.).

2.3.5 Partitionsnamen unter Linux

Die Logik bei der Bezeichnung von Partitionen unter Linux ist zwar klarer als unter Windows, die resultierenden Namen sind aber nicht gerade einprägsam. Statt Laufwerksbuchstaben sind unter Linux so genannte Device-Namen üblich. Normale Festplatten (IDE-Festplatten) werden als `/dev/hd`xy bezeichnet, wobei x ein Buchstabe für die Festplatte ist (`a` für die erste Platte, `b` für die zweite Platte etc.) und y eine Ziffer für die Partition. SCSI-Festplatten werden analog mit `/dev/sd`xy benannt. CD-ROM- und DVD-Laufwerke werden wie Festplatten behandelt. Externe USB- und Firewire-Geräte werden wie SCSI-Geräte behandelt.

Für die Reihenfolge von IDE-Geräten ist die interne Verkabelung entscheidend. `/dev/hda` bezeichnet das erste Gerät (den Master) am ersten IDE-Kanal. `/dev/hdb` bezeichnet das zweite Gerät (den Slave) am ersten IDE-Kanal. `/dev/hdc` und `/dev/hdd` gelten sinngemäß für das Master- und Slave-Gerät am zweiten IDE-Kanal. Es ist durchaus möglich, dass zwei Geräte die Device-Namen `/dev/hda` und `/dev/hdc` bekommen, `/dev/hdb` aber ungenutzt bleibt – nämlich dann, wenn die Geräte jeweils als Master am ersten und am zweiten Kanal angeschlossen sind.

Bei SCSI-Geräten hängt die Reihenfolge von den ID-Nummern der Geräte ab. Löcher in der ID-Reihenfolge werden nicht berücksichtigt. Drei SCSI-Geräte mit den ID-Nummern 0, 2 und 5 bekommen also die Device-Namen `/dev/sda` bis `/dev/sdc`. Wenn nun ein Gerät mit der ID-Nummer 3 hinzugefügt wird, bekommt dieses den Namen `/dev/sdc`; das Gerät mit der ID-Nummer 5 wird jetzt als `/dev/sdd` angesprochen.

Für die Nummerierung der Partitionen gilt – unabhängig von IDE oder SCSI –, dass die Ziffern 1 bis 4 für primäre oder erweiterte Partitionen reserviert sind, die Ziffern ab 5 für logische Partitionen innerhalb der erweiterten Partitionen. Aus diesem Grund kommt es recht häufig vor, dass es in der Nummerierung Löcher gibt. (Beispielsweise bezeichnet `hda1` eine primäre, `hda2` eine erweiterte

und `hda5` dann die erste logische Partition.) Die folgende Tabelle gibt einige Beispiele.

Device-Namen von Festplattenpartitionen

`/dev/hda`	die gesamte erste IDE-Platte
`/dev/hda1`	die erste primäre Partition der ersten IDE-Platte
`/dev/hda2`	die erweiterte Partition der ersten IDE-Platte
`/dev/hda5`	die erste logische Partition der ersten IDE-Platte
`/dev/hda8`	die vierte logische Partition der ersten IDE-Platte
`/dev/hdb`	die gesamte zweite IDE-Platte
`/dev/hdb1`	die erste primäre Partition der zweiten IDE-Platte

Diese Device-Namen benötigen Sie übrigens nur zu Verwaltungszwecken, d. h. wenn Sie die Partitionierung einer Festplatte ändern oder eine bestimmte Partition in das Dateisystem einbinden möchten. Im normalen Betrieb greifen Sie auf das gesamte Dateisystem über Verzeichnisse zu. Dabei bezeichnet `/` den Start des Dateisystems. Einzelne Partitionen können darin an beliebigen Orten eingebunden werden – eine zusätzliche Linux-Partition etwa unter dem Namen `/data`, eine Windows-Partition beispielsweise unter dem Namen `/windows/C`.

Hinweis

Wenn Sie mit Kernel 2.4 oder einer älteren Version arbeiten, gibt es noch einen Sonderfall: Damit das Brennen von CDs bzw. DVDs mit IDE-CD-R- bzw. IDE-DVD-R-Laufwerken möglich ist, wird beim Rechnerstart für derartige Laufwerke meist die Option *ide-scsi* verwendet. (Knoppix verwendet diese Option sogar generell für alle Laufwerke.) *ide-scsi* bewirkt, dass die betreffenden IDE-Laufwerke wie SCSI-Laufwerke behandelt werden. In logischer Konsequenz gilt dies auch für die Device-Namen.

2.4 Neupartitionierung der Festplatte

Dieser Abschnitt geht davon aus, dass auf Ihrem Rechner entweder Windows 9x/ME installiert ist und die Partition die gesamte Festplatte füllt oder Windows NT/2000/XP in einer Partition mit einem NTFS-Dateisystem installiert ist und ebenfalls die Partition die gesamte Festplatte füllt. Das Ziel besteht darin, diese Partition zu verkleinern, sodass anschließend Platz für Linux-Partitionen vorhanden ist. Das ist natürlich nur möglich, wenn die Partition nicht bereits vollständig mit Dateien gefüllt ist!

Bei einer Re-Partitionierung sind Probleme nie auszuschließen. Daher ist eine Sicherung aller wichtigen Daten vor derartigen Aktionen unbedingt anzuraten!

> **Tipp**
> Statt sich in das Umpartitionierungs-Abenteuer zu stürzen, sollten Sie in Betracht ziehen, Ihrem Computer eine zweite Festplatte für Linux zu spendieren. Zum einen ist die vorhandene Platte sowieso schon gut gefüllt und zum anderen kosten Festplatten heute nicht mehr die Welt. Eventuell liegt sogar irgendeine kleinere Platte noch in der heimischen Grabbelkiste rum.

Zur Neupartitionierung unter **Windows 9x/ME** gibt es mehrere Möglichkeiten, die mehr oder weniger komfortabel sind:

- Das mit Windows mitgelieferte Werkzeug `FDISK` erlaubt zwar eine Veränderung der Partitionierung, nicht aber die verlustfreie Verkleinerung einer Partition. Sie müssen daher anschließend Windows neu installieren.
- Bei einigen Distributionen hilft das Linux-Installationsprogramm bei der Verkleinerung. Damit das klappt, müssen vor dem Beginn der Installation unter Windows die Programme `SCANDISK` und `DEFRAG` ausgeführt werden.
- Das auf vielen Linux-CDs vorhandene Programm `FIPS` ermöglicht eine verlustfreie Verkleinerung der Windows-Partition. `FIPS` muss vor Beginn der Linux-Installation ausgeführt werden. Das Programm ist allerdings eher kompliziert zu bedienen.
- Erheblich mehr Komfort bieten kommerzielle Werkzeuge. Am bekanntesten sind PartitionMagic von Norton/Symantec und PartitionExpert von Acronis:

  http://www.norton.com/partitionmagic/
  http://www.acronis.com/products/partitionexpert/

Zur Neupartitionierung unter **Windows NE/2000/XP** gibt es auch mehrere Möglichkeiten:

- Der mit Windows mitgelieferte Disk-Manager erlaubt zwar eine Veränderung der Partitionierung, nicht aber die verlustfreie Verkleinerung einer Partition. Sie müssen daher anschließend Windows neu installieren.
- Bei einigen Distributionen hilft das Linux-Installationsprogramm bei der Verkleinerung (siehe oben).
- Wenn Sie Linux von einer Knoppix-CD oder einem vergleichbaren System starten, können Sie die Windows-Partition mit `ntfsresize` verändern.
- Schließlich besteht die Möglichkeit, die Partitionierung mit einem kommerziellen Tool durchzuführen (siehe oben).

Tipp

Bevor Sie die Partitionierung Ihrer Festplatte ändern, sollten Sie sich natürlich überlegen, wie viel Platz Sie für Windows und Linux reservieren möchten. Wenn Sie eine sehr große Festplatte besitzen und vorerst mehr Platz haben, als Sie in nächster Zeit für Windows und Linux zusammen benötigen, sollten Sie sich eher für kleinere Partitionen entscheiden. Zwar ist es hinterher unmöglich, eine Partition zu vergrößern, es ist aber recht einfach, eine zusätzliche Partition – egal, ob nun für Windows oder für Linux – einzurichten. Sobald aber einmal die gesamte Festplatte mit Partitionen gefüllt ist, verlieren Sie diese Flexibilität.

2.4.1 Neupartitionierung unter Windows 9x/ME

Aktuelle Partitionierung feststellen: Die einfachste Methode besteht darin, unter Windows im Explorer das Laufwerk `C:` mit der rechten Maustaste auszuwählen und das Kommando *Eigenschaften* anzuklicken. In einem Dialog wird nun die Größe der Partition angezeigt. Wenn diese näherungsweise der Festplattengröße entspricht, dann gibt es eine einzige große Partition, die die gesamte Festplatte füllt.

Wenn Sie vermuten, dass es mehrere Partitionen gibt, starten Sie die DOS-Eingabeaufforderung und führen dort das Programm `FDISK` aus. Das Kommando (4) zeigt eine Liste aller für Windows erkennbaren Partitionen an. (Eventuell vorhandene Linux-Partitionen werden übrigens ignoriert.)

Partitionierung mit FDISK ändern: `FDISK` kann auch dazu verwendet werden, um die Partitionierung der Festplatte zu ändern. Es gibt zwar kein eigenes Kommando, um eine Partition zu verkleinern, aber Sie können die Partition löschen und anschließend neu (und kleiner) anlegen. Bei dieser Operation verlieren Sie jedoch alle Daten! Sie müssen also anschließend Windows und alle dort installierten Programme neu aufspielen. Zweitens sollten Sie mit `FDISK` generell nur Partitionen verändern, die momentan nicht in Verwendung sind. Dazu müssen Sie den Rechner mit einer Notfalldiskette starten und `FDISK` von dort ausführen.

Verkleinerung einer Partition durch das Linux-Installationsprogramm: Einige Linux-Distributionen bieten während der Installationsphase die Möglichkeit an, eine vorhandene Windows-Partition zu verkleinern. Die Bedienung der Programme ist in beiden Fällen einfach.

Damit die Verkleinerung der Windows-Partition erfolgreich ist, müssen Sie vor der Linux-Installation Windows starten und dort die Festplatte defragmentieren. Dadurch werden die Dateien im vorderen Teil der Partition gleichsam zusammengeschoben.

Die Defragmentierung ist notwendig, weil die Programme zur Partitionsverkleinerung den hinteren Teil der Partition nur dann gleichsam abschneiden können,

wenn dieser frei von Dateien ist. Vor dem Defragmentieren sollten Sie die Partition auf Fehler untersuchen (z. B. mit `CHKDSK` oder `SCANDISK`) und eventuell vorhandene Sektoren, die keiner Datei mehr zugeordnet sind, löschen. Die verkleinerte primäre Partition kann auf keinen Fall kleiner werden als die darauf gespeicherten Daten (ist ja klar, oder?).

Zum Defragmentieren steht das Programm `DEFRAG` zur Verfügung. Sie können dieses Programm auch im Explorer starten: Klicken Sie das Symbol der Festplatte mit der rechten Maustaste an, wählen Sie *Eigenschaften* → *Extras* → *Jetzt optimieren*. Das Problem bei der Anwendung der Defragmentierprogramme besteht darin, dass sie manche Systemdateien als nicht verschiebbar betrachten. Wenn Sie Pech haben und eine nicht verschiebbare Datei im hinteren Bereich der Partition liegt, gelingt die Verkleinerung der Partition nicht im vollen Ausmaß. Fallen die Ergebnisse der Defragmentierung enttäuschend aus, versuchen Sie `DEFRAG C: /p /f /h`.

Hilft auch das nicht, können Sie die Festplatte nach nicht verschiebbaren Dateien durchsuchen und deren System- und Hidden-Attribute manuell zurücksetzen. Das DOS-Kommando `ATTRIB /S C:\*.* > datei` schreibt eine Liste mit den Attributen aller Dateien der Festplatte in eine Textdatei, die Sie nun bequem mit einem Editor lesen können. Nun ändern Sie bei einzelnen Dateien diese Attribute – mit `ATTRIB` oder im Explorer (rechte Maustaste, Kontextmenü *Eigenschaften*). Verzichten Sie auf Änderungen direkt im Wurzelverzeichnis. Anschließend führen Sie das Defragmentierprogramm ein weiteres Mal aus und hoffen auf bessere Resultate.

Das Suse-Handbuch empfiehlt schließlich, eventuell durchgeführte Optimierungen der Auslagerungsdatei aufzuheben, weil auch diese ein mögliches Hindernis darstellen kann.

Verkleinerung einer Partition durch FIPS: Wenn Sie eine Distribution installieren, deren Installationsprogramm keine Werkzeuge zur Verkleinerung einer Windows-Partition vorsieht, finden Sie im Regelfall das Programm `FIPS.EXE` im Verzeichnis `dosutils` der Linux-CD. Mit diesem Programm können Sie in den meisten Fällen ebenfalls eine vorhandene Windows-Partition verkleinern, ohne dass dabei Daten verloren gehen. Sie finden FIPS natürlich auch unter *http://www.igd.fhg.de/ aschaefe/fips/*. `FIPS` kann nicht mit NTFS-Partitionen von Windows NT/2000/XP umgehen!

`FIPS` teilt eine primäre Partition in zwei kleinere primäre Partitionen. Bei mehreren Festplatten/Partitionen kann sich durch die neue Partition deren Laufwerksbuchstabe ändern. Das gibt sich aber wieder, sobald Sie die neu gewonnene Partition als Linux-Partition einrichten. Die Teilung kann nur im freien Bereich der Partition erfolgen. Aus diesem Grund müssen Sie die Partition vor der Anwendung von `FIPS` wie oben beschrieben defragmentieren.

`FIPS` kann die Partition nicht im laufenden Windows-Betrieb verändern. Sie benötigen eine DOS-Boot-Diskette, mit der Sie den Rechner neu starten, um `FIPS` dann von der Diskette auszuführen.

Um eine Boot-Diskette zu erstellen, legen Sie unter Windows 9x/SE eine leere Diskette ein und führen in einem Kommandoeingabefenster `FORMAT A: /s` aus. Unter Windows ME besteht diese Möglichkeit nicht mehr – erstellen Sie stattdessen in der Systemsteuerung eine so genannte Startup-Diskette. In jedem Fall kopieren Sie anschließend die Programme `FIPS` und `RESTORRB` sowie die Textdatei `error.txt` auf diese Diskette. Mit dieser Diskette starten Sie den Rechner neu.

Die Bedienung von `FIPS` ist dann denkbar einfach: Das Programm untersucht die Festplatte und zeigt das Ergebnis der Analyse an. Die aktuellen Partitionsinformationen (Root- und Bootsektor) können auf die Diskette kopiert werden, um im Fall von Problemen die Partition mit `RESTORRB` später wiederherstellen zu können.

Anschließend können Sie mit den Cursortasten (←) und (→) angeben, an welcher Stelle die Partition in zwei kleinere Partitionen geteilt werden soll. Die Teilungsstelle kann natürlich nur im ungenutzten Bereich der Partition ausgewählt werden. Es ist sinnvoll, auf der zu teilenden Partition zumindest einige MByte frei zu lassen, sonst ist anschließend ein vernünftiges Arbeiten kaum mehr möglich. Nach mehreren Sicherheitsabfragen wird die Partition schließlich tatsächlich geteilt. Sie können das Programm jederzeit mit (Strg)+(C) abbrechen, wenn Sie sich unsicher sind!

Wenn Sie den Rechner nun neu starten, sollte Windows wie bisher funktionieren. Allerdings sollte der freie Plattenplatz jetzt deutlich geringer sein. Dafür besitzen Sie eine neue leere Partition. Starten Sie das Programm `FDISK` und löschen die **neue** Partition. Während der Linux-Installation wird dort eine neue Linux-Partition angelegt.

Zu `FIPS` existiert übrigens eine ausführliche (englische) Online-Dokumentation: `fips.doc` beschreibt die Bedienung des Programms, `errors.txt` enthält eine Liste der möglichen Fehlermeldungen und eine kurze Beschreibung.

2.4.2 Neupartitionierung unter Windows NT/2000/XP

Aktuelle Partitionierung feststellen: Die Windows-Versionen NT/2000/XP wurden mit einem ansprechenden Disk-Manager ausgestattet, der über die Systemsteuerung oder das Administrationsmenü gestartet werden kann (bei jeder Windows-Version ein wenig anders – NT/2000: Festplatten-Manager, XP: Computerverwaltung → Datenträgerverwaltung).

Partitionierung mit dem Windows-Disk-Manager ändern: Mit dem Disk-Manager können Sie Partitionen löschen und neu anlegen, aber nicht verkleinern. Der einzige Weg, eine NTFS-Partition ohne Zusatzsoftware zu verkleinern, führt also zu einem vollständigen Datenverlust und zu einer Neuinstallation von Windows NT/2000/XP.

Partition während der Installation verändern: Bei einigen neueren Linux-Distributionen können NTFS-Partitionen während der Installation verkleinert

werden. Intern wird von beiden Distributionen das unten näher beschriebene Programm `ntfsresize` eingesetzt. Vor dem Beginn der Installation sollte auch hier die Partition unter Windows defragmentiert werden. Sie finden das Windows-Defragmentier-Tool, wenn Sie im Explorer ein Laufwerk mit der rechten Maustaste anklicken und *Eigenschaften* → *Tools/Extras* ausführen.

Partitionierung mit ntfsresize ändern: `ntfsresize` ist ein textorientiertes Linux-Programm, mit dem eine NTFS-Partition ohne Datenverlust verkleinert oder auch vergrößert werden kann. Die Anwendung des Programms ist allerdings relativ kompliziert und daher nur Linux-Profis zu empfehlen. Sie finden dieses Programm samt einer ausführlichen Anleitung im Internet unter *http://linux-ntfs.sourceforge.net/*.

`ntfsresize` kann erst ausgeführt werden, wenn Linux schon läuft. Das tut es aber an dieser Stelle noch nicht. Setzen Sie zur Ausführung von `ntfsresize` eine Linux-Distribution ein, die direkt von der CD läuft und daher nicht installiert werden muss (z. B. Knoppix).

Vor der Verwendung von `ntfsresize` müssen Sie die betroffene NTFS-Partition unter Windows defragmentieren. Um beispielsweise die NTFS-Partition `/dev/hda1` auf 20 GByte zu verkleinern, rufen Sie das Kommando folgendermaßen auf:

```
root#  ntfsresize -s 20G -n /dev/hda1
```

Gelingt das ohne Fehler, wiederholen Sie den Aufruf, aber diesmal ohne die Read-Only-Option `-n`. `ntfsresize` ändert nur die Größe des Dateisystems, nicht aber die der Partitionen! Um die Partition an die neue Dateisystemgröße anzupassen, müssen Sie die Partition mit `fdisk` oder einem anderen Partitionseditor löschen und in der richtigen Größe neu anlegen. Dabei sollten Sie einige Dinge beachten:

- Stellen Sie sicher, dass die neue Partition den gleichen Dateisystemtyp aufweist und denselben Startzylinder verwendet!
- Bei `ntfsresize` wird die gewünschte Größe in k, M oder G angegeben. Diese Buchstaben entsprechen 1000, eine Million bzw. eine Milliarde Byte (nicht 1024 bzw. Vielfachen von 1024).
- War die Partition vorher als *bootable* markiert, sollten Sie dieses Attribut auch bei der neuen Partition wieder setzen.

Damit sind die Vorarbeiten abgeschlossen. Ab jetzt wird vorausgesetzt, dass auf Ihrem Rechner ausreichend Platz für eine oder mehrere Linux-Partitionen ist bzw. dass Sie Ihre Windows-Partition defragmentiert haben, damit ein in das Linux-Installationsprogramm integriertes Partitionsverkleinerungsprogramm funktioniert.

2.5 Start der Linux-Installation

Sie beginnen die Installation damit, dass Sie die Installations-CD oder -DVD in Ihr CD-ROM-Laufwerk legen und den Rechner neu starten. (Auch wenn wir immer von einer Installations-CD bzw. von CD-ROM-Laufwerken schreiben, gelten diese Informationen unverändert auch für DVDs bzw. DVD-Laufwerke.) Statt des üblichen Starts Ihres bereits installierten Betriebssystems sollte nun das Linux-Installationsprogramm direkt von der CD starten. Sollte dies nicht gelingen, kann das mehrere Gründe haben:

- Meist ist Ihr BIOS so konfiguriert, dass ein Booten von einer CD-ROM nicht möglich ist. Um die BIOS-Einstellungen zu ändern, müssen Sie unmittelbar nach dem Einschalten des Rechners eine Taste drücken (häufig (Entf) oder (F1)). Wie Sie in das BIOS-Menü gelangen bzw. wie die Veränderung des BIOS im Detail erfolgt, hängt von Ihrem Mainboard ab und kann daher hier nicht beschrieben werden. Informationen darüber finden Sie im Handbuch zu Ihrem Rechner. Beachten Sie, dass während der BIOS-Einstellung meist das amerikanische Tastaturlayout vorausgesetzt wird ((Y) und (Z) vertauscht).
- Falls Sie ein SCSI-CD-ROM-Laufwerk verwenden, müssen Sie eine äquivalente Einstellung, die das Booten von der CD erlaubt, unter Umständen im BIOS der SCSI-Karte einstellen. In das Konfigurationsmenü gelangen Sie etwas später, wenn am Bildschirm eine Erkennungsmeldung für die SCSI-Karte erscheint. Das erforderliche Tastenkürzel hängt von der SCSI-Karte ab. (Bei vielen Adaptec-SCSI-Karten ist (Strg)+(A) gebräuchlich.)
- Einige sehr alte Rechner erlauben gar kein Booten von einer CD. In solchen Fällen müssen Sie eine Boot-Diskette verwenden. Auf dieser Diskette befindet sich der Beginn des Installationsprogramms, der unter anderem den Zugriff auf das CD-ROM-Laufwerk ermöglicht. Von dort wird dann das restliche Installationsprogramm geladen. Tipps, wie Sie eine derartige Diskette selbst erstellen können, finden Sie auf Seite 59.

Fast alle gängigen Distributionen verwenden ein grafisches Installationsprogramm. Damit dies ausgeführt werden kann, sind ausreichend RAM und eine VGA-kompatible Grafikkarte erforderlich. Auf modernen PCs sind diese Voraussetzungen meistens erfüllt. Je nach Hardware kommt es manchmal leider vor, dass die Grafikkarte nicht korrekt angesteuert wird oder dass der Monitor nach dem Start des Installationsprogramms plötzlich schwarz wird (häufig bei Flachbildschirmen, die per DVI-Kabel an die Grafikkarten angeschlossen sind, weil die Grafikkarte in den Analogmodus wechselt). In solchen Fällen können die meisten Distributionen auch im Textmodus installiert werden. Wie Sie während des Installationsstarts den Text- statt des Grafikmodus auswählen, hängt von der jeweiligen Distribution ab.

Kernel-Meldungen: Bei manchen Distributionen werden unmittelbar nach dem Start eine Menge Meldungen des Kernels angezeigt, die Sie getrost ignorieren können. Nur wenn das Installationsprogramm während dieser Phase hängen bleibt, lassen die Kernel-Meldungen unter Umständen einen Rückschluss darauf zu, was die Ursache des Problems sein könnte.

Erste Einstellungen: Die ersten Einstellungen im Installationsprogramm betreffen die Auswahl der Installationssprache, der Tastatur und eventuell der Maus. Manche Installationsprogramme unterscheiden auch zwischen verschiedenen Installationsmodi (für Einsteiger, für Experten etc.), die sich vor allem dadurch unterscheiden, welche Möglichkeiten es gibt, die Installation wirklich an Ihre Anforderungen anzupassen.

Tastaturauswahl: Viele Konfigurationsprogramme geben mehrere Tastaturen zur Auswahl an, wobei als Entscheidungskriterium manchmal die Anzahl der Tasten gilt. Da Sie wahrscheinlich keine Lust zum Tastenzählen haben, kurz einige Tipps:

Englische Tastatur ohne Windows-Tasten: 101 Tasten
Internationale Tastatur ohne Windows-Tasten: 102 Tasten
Englische Tastatur mit Windows-Tasten: 104 Tasten
Internationale Tastatur mit Windows-Tasten: 105 Tasten

Außerdem können Sie bei manchen Distributionen so genannte *dead keys* wahlweise aktivieren oder deaktivieren. Mit diesen Tasten sind beispielsweise (´) und (˜) gemeint. (De)aktiviert werden nicht die Tasten an sich, sondern das Zusammensetzen von Buchstaben mit diesen Tasten.

Die Aktivierung der *dead keys* bedeutet, dass Sie beispielsweise den Buchstaben Ç in der Form (˜), (C) eingeben können. Wenn Sie die den *dead keys* zugeordneten Buchstaben selbst eingeben möchten, müssen Sie die betreffende Taste und danach die Leerzeichentaste drücken (also (˜), (Leertaste) für ˜).

Die Deaktivierung der *dead keys* bewirkt, dass Sie keine ausländischen Sonderzeichen zusammensetzen können. Dafür ist jetzt die Eingabe der den *dead keys* zugeordneten Zeichen einfacher, weil das betreffende Zeichen sofort erscheint. Wenn Sie keine ausländischen Sonderzeichen eingeben möchten, ist die Deaktivierung also sinnvoller.

Mausauswahl: Im Regelfall wird die Maus automatisch erkannt. Klappt das nicht, müssen Sie aus einer Liste Ihr Modell (oder ein dazu kompatibles) auswählen. Falls Sie eine Maus mit nur zwei Tasten und ohne drückbares Rad besitzen, sollten Sie die Option *Emulate 3 Buttons* auswählen. Damit erreichen Sie, dass Sie die fehlende Maustaste durch das gleichzeitige Drücken beider Tasten simulieren können.

Das Linux-Installationsprogramm läuft selbst bereits unter Linux. Dazu wird unmittelbar nach dem Start zuerst der Linux-Kernel geladen und dann von diesem das eigentliche Installationsprogramm gestartet.

Der Kernel muss beim Start Ihre Hardware erkennen. Es werden zwar nicht alle Hardware-Komponenten benötigt, aber zumindest der Zugriff auf die Festplatten (auch bei SCSI-Systemen), auf das CD- oder DVD-Laufwerk oder auf die Netzwerkkarte muss funktionieren. Nun gibt es leider Dutzende SCSI-Karten und vermutlich Hunderte von Netzwerkkarten, die zueinander inkompatibel sind.

Aus diesem Grund wird vorerst ein Minimal-Kernel geladen, der nicht viel mehr als die IDE-Festplatten erkennt. Alle weiteren Hardware-Komponenten werden durch so genannte Module unterstützt. Das Installationsprogramm versucht selbstständig, die richtigen Module zu aktivieren – und bei den meisten PC-Konfigurationen klappt das tatsächlich automatisch. Nur wenn das nicht der Fall ist, ist es Ihre Aufgabe, die erforderlichen Informationen (natürlich menügesteuert) anzugeben. Einige Installationsprogramme erlauben die Eingabe derartiger Zusatzinformationen nur, wenn Sie die Installation in einem speziellen Modus für Experten bzw. fortgeschrittene Computer-Anwender durchführen.

2.6 Partitionierung der Festplatte unter Linux

Einer der wichtigsten Schritte während der Linux-Installation ist das Anlegen neuer Linux-Partitionen. Nur wenn schon Linux installiert war, können Sie eventuell auch eine schon vorhandene Partition auswählen, ohne die Partitionierung ändern zu müssen. Alle gängigen Distributionen bieten mittlerweile zur Partitionierung der Festplatte einfach zu bedienende grafische Werkzeuge an. Einige Distributionen bieten an, die Festplatte ohne weitere Rückfragen selbst zu partitionieren. Dadurch gelangen Sie aber nicht immer zu optimalen Lösungen.

2.6.1 Anzahl und Größe von Linux-Partitionen

Immer wieder wird mir die Frage gestellt, wie eine Festplatte am besten in Partitionen zerlegt werden soll. Leider gibt es darauf keine allgemein gültige Antwort. Dieser Abschnitt soll Ihnen aber zumindest ein paar Faustregeln für die richtige Anzahl und Größe von Partitionen vermitteln. Sie können Linux (wie Windows) mit einer einzigen Partition betreiben, aber nicht optimal! Besser ist es, den Platz auf mehrere Partitionen zu verteilen:

Systempartition (root-Partition): Sie ist die einzige Partition, die Sie unbedingt benötigen. Sie wird das Linux-System mit all seinen Programmen aufnehmen. Diese Partition bekommt immer den Namen `/`. Dabei handelt es sich genau genommen um den Punkt, an dem die Partition in das Dateisystem eingebunden wird (den `mount`-Punkt). Wenn das System also einmal läuft, sprechen Sie diese Partition mit dem Pfad `/` an. `/` bezeichnet die Wurzel, also den Anfang des Dateisystems. Eine vernünftige Größe für die Installation und den Betrieb einer gängigen Distribution liegt zwischen drei und fünf GByte. Wenn Sie vorhaben, viele Programme auszuprobieren, darf es auch mehr sein.

Boot-Partition: Diese Partition mit dem Namen `/boot` beherbergt lediglich die Daten, die während der ersten Phase des Rechnerstarts erforderlich sind (die Kernel-Datei `vmlinuz`, die Initial-RAM-Disk-Datei `initrd` und die Dateien des Boot-Loaders.). Eine eigene Boot-Partition kann selten auftretende Boot-Probleme vermeiden. Sie sollten sich für eine eigene Boot-Partition entscheiden, wenn eine der beiden folgenden Randbedingungen gilt:

- Sie verwenden ein Mainboard mit einem BIOS-Datum vor 1998. In diesem Fall sind Sie möglicherweise vom 1024-Zylinder-Limit betroffen. Das bedeutet, dass sich alle für das Booten relevanten Dateien innerhalb der ersten 1024 Zylinder der Festplatte befinden müssen. Um dieses Limit zu umgehen, muss sich die Boot-Partition innerhalb dieser Grenze befinden. Der Ort der anderen Partitionen ist dagegen gleichgültig.
- Sie verwenden für die Systempartition ein spezielles Dateisystem, z. B. *reiserfs*, LVM oder RAID. In diesem Fall kann es sein, dass der Boot-Loader nicht in der Lage ist, Dateien von der Systempartition zu lesen. Es ist daher vernünftiger, diese in einer eigenen Boot-Partition unterzubringen.

Home-Partition: Wenn Sie etwas längerfristig denken, ist eine Aufteilung auf zumindest zwei Partitionen – eine System- und eine Datenpartition – vernünftiger. Der Grund: Sie können dann später problemlos eine neue oder andere Distribution in die Systempartition installieren, ohne die davon getrennte Datenpartition mit Ihren eigenen Daten zu gefährden. Bei der Datenpartition wird üblicherweise `/home` als Name (`mount`-Punkt) verwendet.

Swap-Partition: Die Swap-Partition ist das Gegenstück zur Auslagerungsdatei von Windows: Wenn Linux zu wenig RAM hat, lagert es Teile des gerade nicht benötigten RAM-Inhalts dorthin aus. Die Verwendung einer eigenen Partition (statt wie unter Windows einer gewöhnlichen Datei) hat vor allem Geschwindigkeitsvorteile. Linux kann zwar ebenfalls so konfiguriert werden, dass es statt einer Swap-Partition eine Swap-Datei verwendet, das ist aber unüblich und langsam. Im Gegensatz zu den anderen Partitionen erhält die Swap-Partition keinen Namen (keinen `mount-point`), sie wird aus Effizienzgründen direkt angesprochen.

Eine Richtgröße für die Swap-Partition ist das Zweifache der Größe Ihres RAMs. Wenn Sie also 256 MByte RAM besitzen, sollten Sie die Swap-Partition mit etwa 512 MByte einrichten. (Eine größere Swap-Partition ist zwar möglich, aber für die meisten Anwendungsfälle nicht mehr sinnvoll. Wenn Ihre Anwendungen tatsächlich so viel Speicher benötigen, ist Linux nur noch mit der Übertragung von Seiten zwischen der Swap-Partition und dem RAM beschäftigt und praktisch nicht mehr bedienbar. Abhilfe schafft hier nicht eine größere Swap-Partition, sondern mehr RAM.)

Die Aufteilung der Festplatte in Partitionen lässt sich noch viel weiter treiben. Das setzt allerdings etwas Systemkenntnis voraus und ist nur sinnvoll, wenn Linux für spezielle Zwecke, etwa als Netzwerk-Server für ein größeres System verwendet werden soll.

2.6.2 Welches Dateisystem?

Linux unterstützt eine Menge unterschiedlicher Dateisysteme, unter anderem *ext2*, *ext3*, *reiserfs*, *xfs* (SGI) und `jfs` (IBM). Alle Dateisysteme mit der Ausnahme von *ext2* unterstützen Journaling-Funktionen, bieten also größtmögliche Datensicherheit bei einem unvorhergesehenen Absturz (z. B. aufgrund eines Stromausfalls).

Nicht jede Distribution unterstützt alle Dateisysteme während der Installation. Wenn Sie die Wahl haben, sollten Sie sich als Linux-Einsteiger für *ext3* oder *reiserfs* entscheiden. Beide Dateisysteme sind sehr ausgereift und stabil. *reiserfs* ist unter manchen Umständen schneller und platzsparender, dafür gilt *ext3* aber als der De-facto-Standard für Linux-Dateisysteme.

Manche Distributionen bieten während der Installation auch die Möglichkeit, ein LVM- oder RAID-System einzurichten. LVM (*Logical Volume Manager*) ermöglicht es, virtuelle Partitionen anzulegen und diese im laufenden Betrieb zu vergrößern oder zu verkleinern. Durch RAID (*Redundant Array of Inexpensive/Independent Disks*) werden die Partitionen mehrerer Festplatten miteinander verknüpft, um auf diese Weise ein zuverlässigeres und/oder schnelleres Gesamtsystem zu erreichen. LVM und RAID haben aber den Nachteil, dass sie die Administration erschweren. Sie sind für Linux-Einsteiger definitiv nicht geeignet.

In der Swap-Partition wird *kein* richtiges Dateisystem eingerichtet! Die Partition muss vor der ersten Verwendung aber dennoch (mit `mkswap`) formatiert werden. Alle verbreiteten Linux-Distributionen kümmern sich automatisch darum. Dateisystemempfehlung für den Heimgebrauch:

`/boot`	*ext2*
`[swap]`	kein Dateisystem erforderlich
`/`	*ext3* oder *reiserfs*
`/home`	*ext3* oder *reiserfs*

2.7 Paketauswahl

Ein wichtiger Teil der Installation ist die Auswahl des Installationsumfangs. Es ist selten sinnvoll, einfach alles zu installieren:

- Der riesige Umfang der verfügbaren Software-Pakete überfordert gerade Einsteiger. Erheblich übersichtlicher ist es, vorerst nur eine Grundinstallation durchzuführen und die benötigten Zusatzprogramme später bei Bedarf nachzuinstallieren.
- Es gibt Programme, die sich gegenseitig im Weg sind. So können Sie beispielsweise auf einem Rechner nicht zwei E-Mail-Server gleichzeitig betreiben.

- Wenn Sie vorhaben, den Rechner als Netzwerk-Server einzusetzen, vergrößert jedes installierte Programm die potenziellen Sicherheitsrisiken.
- Schließlich bleibt Ihnen mehr Platz für Daten.

Die Auswahl der Software-Pakete erfolgt bei jeder Distribution anders (zumeist in vorkonfigurierten Gruppen).

2.7.1 Grundkonfiguration

Etwas verwirrend ist in manchen Fällen die vom Installationsprogramm anschließend durchgeführte Konfiguration des Systems. Fallweise werden Sie dabei nach Details gefragt, die Sie vielleicht noch nicht kennen oder abschätzen können. Verschieben Sie Ihnen unverständliche Installationsschritte auf später! Bei fast allen Distributionen können Sie entweder das Installationsprogramm selbst oder vergleichbare Werkzeuge auch im laufenden Betrieb ausführen, um notwendige Konfigurationsschritte nachträglich durchzuführen bzw. zu verändern.

Root-Passwort, Benutzerverwaltung: Unter Linux ist für die Systemverwaltung der Benutzer `root` vorgesehen. Dieser Benutzer hat uneingeschränkte Rechte, aber natürlich ist damit auch das Schadenspotenzial uneingeschränkt. Der Zugang zu `root` muss daher mit einem Passwort abgesichert werden.

Es ist unter Linux unüblich, als `root` zu arbeiten (außer natürlich bei der Durchführung administrativer Aufgaben). Wenn Sie einen Brief schreiben oder im Internet surfen, melden Sie sich als gewöhnlicher Benutzer an. Bereits während der Installation haben Sie die Möglichkeit, einen oder mehrere derartige Benutzer samt Passwort einzurichten. Selbstverständlich können Sie später weitere Benutzer hinzufügen, das Passwort vorhandener Benutzer verändern etc.

Hinweis

Linux-Benutzernamen bestehen aus maximal acht Buchstaben und Ziffern. Verwenden Sie besser keine deutschen Sonderzeichen. (Das funktioniert zwar meistens, aber nicht immer.) Es ist üblich, nur Kleinbuchstaben zu verwenden – aber das ist keine Bedingung.
Das Passwort sollte mindestens sechs bis maximal acht Zeichen lang sein. Idealerweise enthält es sowohl Groß- als auch Kleinbuchstaben sowie mindestens eine Ziffer. Auch diverse Sonderzeichen sind erlaubt, z. B. `+-*/_.,;:()[]`. Deutsche Sonderzeichen (äöüß) und andere Buchstaben, die nicht im ASCII-Zeichensatz definiert sind, sollten Sie hingegen vermeiden.

Grafikkarte, Monitor: Die meisten Installationsprogramme konfigurieren auch das für die grafische Ausgabe zuständige X Window System. Kritisch sind hier die Erkennung der Grafikkarte sowie die korrekte Angabe der Monitordaten (maximale Auflösung, maximale Bildfrequenz). Im Regelfall können Sie den

für Sie optimalen Grafikmodus (Bildschirmauflösung, Farbtiefe) auswählen und gleich ausprobieren.

Im Rahmen der Grafikkonfiguration werden Sie manchmal auch gefragt, wie das System in Zukunft gestartet werden soll: im Textmodus oder im Grafikmodus.

- **Grafikmodus:** Diese Variante ist komfortabler und eleganter. Unmittelbar nach dem Systemstart erscheint ein grafischer Login-Dialog, in dessen Folge KDE oder Gnome ausgeführt wird. (Bereitet der automatische Start des Grafiksystems Probleme, führt die Tastenkombination Strg+Alt+F1 in eine Textkonsole.)
- **Textmodus:** Sind Sie im Zweifel, ob die Grafikkonfiguration geklappt hat, ist es sicherer, wenn Sie sich für den Textmodus entscheiden. In diesem Fall müssen Sie sich nach dem Systemstart zuerst im Textmodus einloggen und das Grafiksystem dann manuell mittels `startx` starten.

Netzwerkkonfiguration: Ganz egal, ob Sie Linux in einem lokalen Netzwerk verwenden oder nicht, werden automatisch die elementaren Netzwerkfunktionen installiert. Diese werden nämlich auch für die Linux-interne Kommunikation benötigt. („Loopback-Netzwerk“)

Wollen Sie Ihren Rechner als Client in einem lokalen Netz einzusetzen, müssen Sie außerdem eine Netzwerkkonfiguration durchführen. Diese Konfiguration funktioniert bei vielen Installationsprogrammen vollautomatisch, wenn sich im lokalen Netz ein so genannter DHCP-(Dynamic Host Configuration Protocol) Server befindet, der allen anderen Rechnern im Netzwerk automatisch die Netzwerkparameter sendet. In diesem Fall reduziert sich die gesamte Netzwerkkonfiguration auf das Anklicken der entsprechenden Option.(Eventuell müssen Sie noch den Rechnernamen angeben.) Bei einer manuellen Netzwerkkonfiguration werden Sie nach den folgenden Parametern gefragt:

- **Host- und Domainname:** Der Host- und der Domainname entsprechen unter Windows dem Rechnernamen und dem Workgroup-Namen. In einem lokalen Netz ist der Domain-Name meist vorgegeben. Der Host-Name sollte eindeutig sein. Verwenden Sie als Host-Namen nicht *localhost*, dieser Name hat eine besondere Bedeutung!
- **IP-Adresse des Rechners:** Diese Zahl in der Form a.b.c.d (z. B. 192.168.27.35) dient zur internen Identifizierung des Rechners im Netz. Üblicherweise sind die drei ersten Zahlengruppen bereits durch das lokale Netz vorgegeben (z. B. 192.168.27); die vierte Zahl muss innerhalb des Netzes eindeutig sein und darf weder 0 noch 255 sein.
- **Netzwerkmaske, Netzwerk- und Broadcast-Adresse:** Die Ausdehnung eines lokalen Netzes wird durch zwei oder drei Masken ausgedrückt, die hier ganz kurz anhand eines Beispiels erläutert werden: Wenn das lokale Netz alle Nummern 192.168.27.n umfasst, lautet die dazugehörige Netzwerkmaske

255.255.255.0. Als Netzwerkadresse ergibt sich 192.168.27.0, als Broadcast-Adresse 192.168.27.255.

- **Gateway-Adresse:** Wenn es im lokalen Netz einen Rechner gibt, der für alle anderen Rechner den Internet-Zugang herstellt, dann geben Sie dessen IP-Adresse an.
- **Nameserver-Adresse:** Der so genannte Nameserver (oft auch DNS für Domain Name Service) ist für die Auflösung von Netzwerknamen in IP-Adressen zuständig. Beim Nameserver kann es sich wahlweise um ein lokal laufendes Serverprogramm, einen Rechner im lokalen Netz (wenn er auch für die Auflösung lokaler Namen zuständig ist) oder um einen externen Rechner des Internet Service Providers handeln. Sind Gateway- und Nameserver-Adresse korrekt angegeben, ist Ihr Rechner damit bereits ans Internet angeschlossen.

Modem, ISDN-Karte, DSL, Internet-Konfiguration: Wenn Ihr Rechner nicht Teil eines lokalen Netzes ist, wollen Sie wahrscheinlich per Modem, ISDN-Karte oder DSL ins Internet. Dies gelingt nur in einfachen Fällen auf Anhieb. Deswegen ist es meist sinnvoller, die Internet-Konfiguration auf später zu verschieben.

Drucker, Scanner, Soundkarte: Bei manchen Distributionen können Sie bereits während der Installation den Drucker, die Soundkarte und eventuell andere Hardware-Komponenten konfigurieren. Wenn das problemlos gelingt, spricht nichts dagegen; wenn es aber Probleme gibt, verschieben Sie die Konfiguration besser auf später.

2.7.2 Installation des Boot-Loaders

Die letzte Frage ist nun noch, wie Linux in Zukunft gestartet werden soll. Dazu wird bei den meisten Distributionen das Programm GRUB (manchmal auch LILO) eingesetzt. Es kann wahlweise auf eine Boot-Diskette oder direkt auf die Festplatte installiert werden.

Installation auf eine Diskette: Eine Installation auf eine Diskette bedeutet, dass sich diese Diskette bei jedem Linux-Start im Diskettenlaufwerk Ihres Rechners befinden muss. Ist das nicht der Fall, wird wie bisher Windows gestartet. Der Vorteil einer Boot-Diskette besteht darin, dass die Installation absolut sicher ist: Der bisher vorhandene Boot-Prozess Ihres bisherigen Betriebssystems wird nicht angerührt, Sie können sich absolut sicher sein, dass dieses weiterhin ohne Probleme starten kann. Leider hat das Verfahren einige Nachteile: Manche Rechner besizen gar kein Diskettenlaufwerk und nicht alle Distributionen unterstützen die Erstellung einer Boot-Diskette. Außerdem muss immer die Boot-Diskette im Diskettenlaufwerk stecken und im BIOS die richtige Boot-Reihenfolge eingestellt sein. Schließlich wird der Bootvorgang um

ein paar Sekunden verlängert, weil Daten von einer Diskette langsamer als von einer Festplatte gelesen werden.

Zur GRUB-Installation auf eine Diskette benötigen Sie während der Installation eine leere, formatierte Diskette. Verwenden Sie als Boot-Diskette möglichst eine frisch formatierte Diskette! Zwar sollte das Installationsprogramm die Diskette gegebenenfalls automatisch selbst formatieren, aber das funktioniert nicht immer zuverlässig.

Installation auf eine Festplatte: Die andere Variante besteht darin, GRUB in den Bootsektor der Festplatte zu installieren. (Dieser Bootsektor wird auch Master Boot Record oder kurz MBR genannt.) Die meisten Installationsprogramme bieten diese Option per Default an.

Der Nachteil der MBR-Installation besteht darin, dass dabei der bisher vorhandene Bootsektor, der meist von Windows stammt, überschrieben wird. Damit ist in Zukunft GRUB nicht nur für den Linux-Start verantwortlich, sondern auch für das richtige Verzweigen in den Windows-Boot-Loader. Wenn bei der GRUB-Installation etwas schief geht, können Sie Windows nicht mehr starten. Mit einer Windows-Boot-Diskette und dem Programmaufruf `FDISK /MBR` kann man in der Regel den Windows-Boot-Loader reparieren.

Damit endet bei den meisten Distributionen der Installationsvorgang. Das frisch installierte System wird nun sofort gestartet. Bei einigen Distributionen ist dazu ein Rechnerneustart erforderlich, bei anderen nicht.

Rechnerstart: Wenn alles geklappt hat, sollte in Zukunft bei jedem Rechnerstart ein Menü (bzw. eine Eingabeaufforderung) erscheinen, in dem Sie das gewünschte Betriebssystem auswählen können. Wenige Sekunden später sollte Linux laufen. Je nachdem, welche Distribution Sie verwenden, kann es sein, dass automatisch nochmals das Installationsprogramm erscheint, sodass Sie nun weitere Konfigurationseinstellungen durchführen können. Ansonsten können Sie damit beginnen, Linux zu erkunden.

2.8 Installationsvarianten

Dieser Abschnitt fasst Informationen über einige Installationsformen zusammen, die in der Praxis seltener auftreten – etwa dann, wenn die Installation auf einem sehr alten Rechner, auf einem Notebook etc. durchgeführt wird.

2.8.1 Installationsdisketten erzeugen

Installationsdisketten sind dann erforderlich, wenn ein direkter Start des Installationsprogramms von der CD unmöglich ist. Zur Erstellung der Installationsdiskette(n) müssen Sie eine Datei von der CD-ROM auf eine leere Diskette kopieren. Bei manchen Distributionen benötigen Sie zwei oder sogar noch mehr Disketten.

Zum Kopieren der so genannten Image-Dateien auf die Diskette(n) können Sie aber nicht einfach `COPY` (Windows) oder `cp` (Linux) verwenden, sondern müssen mit `RAWRITE` (Windows) oder `dd` (Linux) arbeiten! Der Grund: Bei den Image-Dateien handelt es sich nicht um normale Dateien, sondern um ein Abbild eines ganzen Dateisystems. Dieses muss blockweise auf die Diskette kopiert werden.

Das Beschreiben der Disketten unter Windows erfolgt mit `RAWRITE.EXE`. Dieses Programm überträgt den Inhalt einer Datei Sektor für Sektor auf eine formatierte 3,5-Zoll-Diskette. Das Programm (bzw. dessen neuere Version `RAWRITE2.EXE`) befindet sich normalerweise im Verzeichnis `DOSUTILS` der Linux-CD. Dazu ein Beispiel. Das folgende Kommando erzeugt eine Linux-Boot-Diskette (A:) von der CD-ROM (E:) aus:

```
E:\DOSUTILS\RAWRITE -f E:\images\boot.img -d A:
```

Falls Sie bereits ein Linux-System besitzen, können Sie die Installationsdisketten auch dort erzeugen. `/mnt/cdrom` ist der Ort, an dem die CD in das Dateisystem eingebunden ist, und kann je nach Distribution variieren.

```
root#  dd if=/mnt/cdrom/images/boot.img of=/dev/fd0
```

2.8.2 Linux über eine Netzwerkverbindung installieren

Wenn die Installationsdaten weder von einem CD-ROM-Laufwerk noch von der lokalen Festplatte geladen werden können, besteht noch die Möglichkeit einer Netzwerkinstallation. Die Installationsdateien müssen sich also auf einem anderen Rechner im lokalen Netzwerk befinden. Zur Installation gibt es je nach Distribution bis zu vier Varianten:

- FTP (*File Transfer Protocol*): Die Installationsdaten werden von einem FTP-Server gelesen. Diese Variante bietet sich vor allem an Universitäten an, wenn sich die gewünschte Linux-Distribution auf dem Server des örtlichen Rechenzentrums befindet.
- HTTP (*Hypertext Transfer Protocol*): Die Installationsdaten werden von einem Webserver gelesen.
- NFS (*Network File System*): Die Installationsdaten werden von einem anderen Unix- oder Linux-Rechner gelesen. Der Partnerrechner muss dazu als NFS-Server konfiguriert sein; das Verzeichnis mit den Installationsdaten muss via NFS zugänglich sein.
- SMB (*Server Message Block*): Diese Variante ist mit NFS vergleichbar, allerdings wird statt NFS das unter Windows übliche Protokoll SMB verwendet.

Alle Varianten setzen voraus, dass Ihr Rechner mit einer Netzwerkkarte ausgestattet und korrekt verkabelt ist. Außerdem muss bereits während der Installation die Netzwerkkarte erkannt und in Betrieb genommen werden. Dabei müssen dieselben Informationen wie bei der Netzwerkkonfiguration angegeben werden. Darüber hinaus sind zumeist die folgenden Zusatzinformationen erforderlich:

- **Name oder Adresse des FTP- oder NFS-Servers:** Das ist der Rechner, von dem die Daten gelesen werden.
- **Mount-Punkt bzw. FTP-Verzeichnis:** Damit geben Sie den Pfad zu den Installationsdaten an (beispielsweise `/mnt/cdrom` bei einem NFS-Server).
- **User-Name und Passwort:** Diese Angaben können entfallen, wenn der FTP- oder NFS-Server jedem den Zugriff erlaubt (was jedoch selten der Fall ist).

Sobald der Zugriff auf die Installationsdaten gelingt, erfolgt die weitere Installation wie von der CD-ROM. Bei vielen Distributionen müssen Sie zuerst eine oder zwei Installationsdisketten erstellen, bevor Sie mit der Installation über's Netz beginnen können.

2.8.3 Notebook-Installation

Die Installation von Linux auf ein Notebook kann eine besondere Herausforderung darstellen – und zwar aus mehreren Gründen:

- Das Energie-Management, also die Stromsparmodi, das Ein- und Ausschalten von Komponenten etc., wird über Funktionen des ACPI-Standard gesteuert. (ACPI steht für *Advanced Configuration and Power Interface*.) Linux unterstützt ACPI zwar prinzipiell, in der Praxis treten dabei aber oft Probleme auf.
- Bei manchen sehr kleinen Notebooks steht kein CD- oder DVD-ROM-Laufwerk zur Verfügung. In diesem Fall müssen Sie ein externes CD-Laufwerk anschließen oder eine Netzwerkinstallation durchführen (siehe oben).
- Die in Notebooks eingebauten Modems sind meist so genannte WinModems, von denen viele Modelle unter Linux nicht genutzt werden können.
- Auch zu den in Notebooks eingebauten WLAN-Karten fehlen zum Teil noch Linux-Treiber. Die einfachste Abhilfe ist hier eine PCMCIA-Steckkarte für den etwas langsameren WLAN-Standard 802.11b, der bereits sehr gut unterstützt wird.

- Die in Notebooks eingesetzten Grafikchips werden zum Teil nur von ganz aktuellen XFree86-Versionen unterstützt. Sofern der Bildschirm eine Standardauflösung hat, kann die Grafikkarte aber zumeist im VESA- oder Framebuffer-Modus benutzt werden.

Ein fast unerschöpfliches Angebot an Informationen zum Thema Linux und Notebooks samt Links zu den verwandten Themen PCMCIA, WinModems etc. finden Sie im Internet. Lesen Sie diese Seiten möglichst schon, bevor Sie sich Ihr Notebook kaufen! Achten Sie auch bei der Auswahl Ihrer Distribution darauf, dass Notebook-Installationen unterstützt werden. Das ist am ehesten bei weit verbreiteten Distributionen der Fall. Manchmal gibt es schon Notebooks mit fertig installiertem Linux zu kaufen.

2.9 Probleme bei der Installation

Dieser Abschnitt geht auf einige typische Probleme ein, die während der Installation auftreten können. So weit möglich, finden Sie hier auch Lösungsansätze.

Was tun Sie, wenn es während der Installation Probleme gibt, der Rechner stehen bleibt, Hardware nicht oder falsch erkennt etc.? Der erste Tipp ist geradezu trivial: Lesen Sie vor Beginn der Installation auf jeden Fall die `README`-Dateien bzw. die so genannten Release-Notes auf der CD-ROM!

Ein weiterer guter Startpunkt sind natürlich die Homepages der jeweiligen Distributionen. Dort gibt es eigene Support-Bereiche (Support-Datenbank bei Suse, Knowledge Base bei Red Hat), wo sich oft Antworten zu häufigen Fragen bzw. Problemen finden:

http://www.mandriva.com
http://www.redhat.com bzw. *http://www.redhat.de*
http://www.Suse.de

Ebenfalls eine exzellente und meist aktuellere Informationsquelle sind die diversen Linux-Newsgruppen. Bei der Suche nach Informationen verwenden Sie am besten *http://groups.google.com.* Wenn Sie beispielsweise als Suchbegriffe „linux installation hardwarexy“ verwenden, werden Sie unweigerlich auf eine ganze Menge Diskussionsbeiträge stoßen, die über Probleme mit dieser speziellen Hardware-Komponente berichten. Mit etwas Glück und Geduld finden Sie dort auch eine Lösung für Ihr Problem.

2.9.1 Hardware-Probleme

Was tun Sie, wenn Linux für die Installation wichtige Hardware-Komponenten nicht richtig erkennt oder beim Erkennungsversuch hängen bleibt?

Der Linux-Kernel kennt einen Mechanismus, mit dem beim Start Parameter übergeben werden können, die bei der Erkennung von Hardware helfen. Die

Übergabe dieser Parameter sieht allerdings bei jeder Linux-Distribution ein wenig anders aus. Zumeist können die Parameter beim Installationsstart angegeben werden. (Dazu muss unter Umständen auf die grafische Installation verzichtet werden.)

Weitere Informationen zu diesem Mechanismus und einen Überblick über einige wichtige bzw. häufig benötigte Optionen erhalten Sie mit `man 7 bootparam` sowie im BootPrompt-HOWTO (*http://www.tldp.org/HOWTO/BootPrompt-HOWTO.html*). Falls der Linux-Kernel-Code installiert ist, sollten Sie einen Blick in die folgenden Dateien werfen:

```
/usr/src/linux/Documentation/kernel-parameters.txt
/usr/src/linux/Documentation/ide.txt
/usr/src/linux/Documentation/*
```

Boot-Optionen werden direkt an den Linux-Kernel übergeben. Sie werden deswegen oft auch Kernelparameter genannt. Die Eingabe derartiger Optionen erfolgt im Boot-Loader GRUB, also unmittelbar nach dem Rechnerstart. Dazu drücken Sie zuerst (Esc), um vom GRUB-Grafikmodus in den Textmodus zu wechseln. Dann wählen Sie mit den Cursortasten die zu startende Linux-Distribution aus. Mit (E) gelangen Sie in den GRUB-Editor, der einige Zeilen angezeigt, die so ähnlich wie das folgende Muster aussehen:

```
kernel (hd0,11)/boot/vmlinuz root=/dev/hda12 splash=silent vga=normal
initrd (hd0,11)/boot/initrd
```

Wählen Sie mit den Cursortasten die *kernel*-Zeile aus und drücken Sie abermals (E), um diese Zeile zu verändern, und fügen Sie an das Ende dieser Zeile die zusätzlichen Boot-Optionen an. Mit (↵) bestätigen Sie die Änderung. (Esc) führt zurück zum Boot-Menü, wo Sie Linux dann starten. (Die Änderung an den Kernel-Parametern gilt nur für dieses eine Mal, sie wird also nicht bleibend gespeichert.)

Boot-Optionen werden in der Form `option1=wert1 option2=wert2,wert3` angegeben. Beachten Sie, dass Sie keine Leerzeichen angeben dürfen, wenn Sie einer Option mehrere Werte zuweisen möchten (wie für *option2*).

2.9.2 Das 1024-Zylinder-Problem

Der Start von Linux kann Probleme bereiten, wenn Sie Linux auf einem sehr alten Rechner installieren (Mainboard-Datum vor ca. 1998). In diesem Fall gelingt ein Start von Linux nur, wenn sich alle zum Booten erforderlichen Dateien innerhalb der ersten 1024 Zylinder der Festplatte befinden. In den meisten Fällen bedeutet das, dass sich die Dateien innerhalb der ersten 7,9 GByte befinden müssen. Je nach Zylindergröße kann der zulässige Datenbereich aber auf bis zu 504 MByte sinken.

Das 1024-Zylinder-Problem betrifft nur den Linux-Start. Sobald Linux einmal läuft, kann es die gesamte Festplatte ansprechen. Zur Umgehung des 1024-Zylinder-Limits ist es erforderlich, dass sich die Boot-Daten innerhalb der ersten 1024 Zylinder befinden. Die einfachste Lösung besteht daher darin, dass Sie bei der Partitionierung der Festplatte eine kleine Partition vorsehen, die vollständig unterhalb dieser Grenze liegt. Der Name (Mount-Punkt) für diese Partition muss `/boot` lauten. Im Verzeichnis `/boot` werden unter Linux alle für den Boot-Prozess erforderlichen Dateien gespeichert. Dabei handelt es sich nur um wenige Dateien. Daher reicht für die Boot-Partition eine Größe von ca. 20 MByte vollkommen aus.

2.9.3 Tastaturprobleme

In den ersten Phasen der Installation kann es vorkommen, dass noch kein deutscher Tastaturtreiber installiert ist und daher das amerikanische Tastatur-Layout gilt. Das trifft meistens auch während des Starts des Boot-Loaders zu. Solange der Rechner glaubt, dass Sie mit einer US-Tastatur arbeiten, Sie aber tatsächlich ein deutsches Modell verwenden, sind (Y) und (Z) vertauscht; außerdem bereitet die Eingabe von Sonderzeichen Probleme.

Die folgende Tabelle zeigt, wie Sie diverse Sonderzeichen auf einer deutschen Tastatur trotz eines fehlenden Tastatur-Treibers eingeben können. Dabei wird in der ersten Spalte die auf einer deutschen Tastatur erforderliche Tastenkombination angegeben, um das Zeichen in der zweiten Spalte zu erzeugen. Verwenden Sie auch den numerischen Tastaturblock – die dort befindlichen Sonderzeichen funktionieren mit Ausnahme des Kommas problemlos! Im Verlauf des Installationsprozesses können Sie dann die richtige Tastaturtabelle auswählen.

Tastenkürzel zur Eingabe von Sonderzeichen für das US-Tastatur-Layout					
(Z)	Y	(Ö)	;	(Shift)+(9)	(
(Y)	Z	(Shift)+(Ö)	:	(Shift)+(0)	)
(-)	/	(Shift)+(-)	?	(Ü)	[
(#)	\	(Shift)+(Ä)	"	(+)	]
(ß)	- (Minus)	(Ä)	'	(Shift)+(Ü)	{
(Shift)+(ß)	_ (Unterstr.)	(^)	`	(Shift)+(+)	}
(')	=	(Shift)+(^)	~	(Shift)+(,)	<
(Shift)+(')	+	(Shift)+(2)	@	(Shift)+(.)	>
(Shift)+(8)	*	(Shift)+(3)	#		
(Shift)+(7)	&	(Shift)+(6)	^		

2.10 Probleme nach der Installation

Manchmal kommt es vor, dass die Installation problemlos gelingt, dass aber anschließend (d. h. beim ersten Neustart) Probleme auftreten. Dieser Abschnitt gibt einige Tipps zu häufigen Problemquellen.

2.10.1 Der Rechner kann nicht mehr gestartet werden

Der *worst case* bei einer Linux-Installation besteht darin, dass anschließend der Rechner gar nicht mehr gestartet werden kann oder dass zumindest einzelne der installierten Betriebssysteme nicht mehr zugänglich sind. Die folgende Liste zählt einige mögliche Varianten auf:

Linux wird gestartet, stürzt aber ab: Nach dem Neustart des Rechners erscheinen zuerst diverse Meldungen von Linux. Anschließend bleibt der Rechner (mit oder ohne Fehlermeldung) stehen bzw. stürzt ab. Die wahrscheinlichste Ursache sind Hardware-Probleme.

Abhilfe: Durch die Angabe von so genannten Boot-Optionen können Sie Linux, d. h. genau genommen dem Kernel, bei der Erkennung der Hardware helfen. Das funktioniert allerdings nur für die Hardware-Komponenten, die unmittelbar vom Kernel angesprochen werden (und nicht von später eingebundenen Kernel-Modulen). Siehe oben unter „Hardware-Probleme".

Wenn die Hardware-Probleme durch Kernel-Module verursacht werden, bleiben die Boot-Optionen wirkungslos. Stattdessen muss die Datei `/etc/modules.conf` (bis Kernel 2.4) oder `/etc/modprobe.conf` (ab Kernel 2.6) geändert werden. Dazu müssen Sie eventuell ein so genanntes Rescue-System (Rettungssystem, Notfallsystem) starten, das sich bei vielen Distributionen auf der Installations-CD befindet. Der Umgang mit einem derartigen System erfordert allerdings einiges an Linux-Wissen und scheidet daher für Linux-Einsteiger aus.

Linux-Absturz mit der Meldung ***kernel panic: unable to mount root fs*****:** Der Start des Linux-Kernels hat geklappt, Linux konnte aber anschließend die Linux-Systempartition nicht finden. Wahrscheinlich war die GRUB-Konfiguration nicht erfolgreich. Das Problem kann aber auch dann auftreten, wenn die Verkabelung der Festplatten geändert wurde.

Abhilfe: Geben Sie beim Linux-Start die richtige Partition als Bootoption in der Form `root=/dev/hdb8` an. Wenn der Start so gelingt, können Sie unter Linux GRUB neu konfigurieren bzw. eine neue Boot-Diskette erstellen. Falls sich die Namen der Partitionen geändert haben (wegen der Neuverkabelung der Festplatten), müssen Sie auch die Datei `/etc/fstab` entsprechend anpassen.

Linux wird nicht gestartet: Nach dem Neustart des Rechners wird ohne Rückfrage einfach Windows gestartet. Von Linux ist keine Spur zu sehen. Die Installation von GRUB (oder eines anderen Boot-Loaders) auf die Festplatte hat aus irgendeinem Grund nicht funktioniert.

Abhilfe: Starten Sie Linux mit der Boot-Diskette. Anschließend versuchen Sie im laufenden System, GRUB neu zu installieren.

Linux kann auch mit der Boot-Diskette nicht gestartet werden: Der Rechner ignoriert die Boot-Diskette beim Rechnerstart. (Das Lämpchen des Diskettenlaufwerks leuchtet nicht auf.) Das BIOS Ihres Rechners ist so konfiguriert, dass in jedem Fall direkt von der Festplatte (oder vom CD-ROM-Laufwerk) gebootet wird, ganz egal, ob sich eine Diskette im Rechner befindet oder nicht.

Abhilfe: Verändern Sie die BIOS-Einstellung für den Rechnerstart. Falls das BIOS korrekt eingestellt ist, ein Booten von der Diskette aber dennoch nicht möglich ist (diverse Fehlermeldungen, Absturz etc.) oder wenn es gar keine Boot-Diskette gibt, können Sie in vielen Fällen auch die Installations-CD zum Start Ihres Linux-Systems verwenden.

Mit der Installations-CD sollte es also gelingen, Linux zu starten. Es kann allerdings sein, dass während des Starts diverse Fehlermeldungen angezeigt werden und dass im Betrieb manche Funktionen nicht zur Verfügung stehen. Der Grund besteht darin, dass einzelne Kernel-Module mit Erweiterungsfunktionen nicht gefunden werden. Dennoch sollten die verbleibenden Funktionen ausreichen, um die GRUB-Konfiguration zu reparieren oder um eine neue Boot-Diskette zu erstellen.

Windows ist nicht mehr zugänglich: Nach dem Neustart wird automatisch Linux gestartet. Windows scheint verschwunden zu sein. Wahrscheinlich hat die GRUB-Installation funktioniert. Sie können nun unmittelbar nach dem Rechnerstart auswählen, welches Betriebssystem gestartet werden soll. Tun Sie nichts, wird nach einer Weile automatisch Linux gestartet.

Abhilfe: Falls ein Menü angezeigt wird, wählen Sie mit den Cursortasten `windows` aus und drücken Sie ↵. Falls es kein Menü gibt, starten Sie Linux und verändern die GRUB-Konfiguration.

Weder Linux noch Windows können gestartet werden: Nach dem Rechnerstart wird GRUB ausgeführt, stürzt aber sofort ab bzw. zeigt eine endlose Liste von Fehlermeldungen an. Die GRUB-Installation ist vermutlich fehlgeschlagen.

Abhilfe: Soweit vorhanden, verwenden Sie eine Boot-Diskette zum Start von Linux. Anschließend versuchen Sie, GRUB neu zu konfigurieren bzw. zu installieren.

Eine andere Möglichkeit besteht darin, den früheren Zustand des Bootsektors (MBR) wieder herzustellen. Wenn das gelingt, kann danach Windows wieder normal gestartet werden. Zum Start von Linux benötigen Sie allerdings eine Boot-Diskette oder eine Installations-CD. Anschließend können Sie einen neuen Versuch starten, den Linux-Boot-Loader zu installieren.

Die Vorgehensweise zur Wiederherstellung des MBR hängt von der Windows-Version ab:

- Für Windows 9x/ME benötigen Sie dazu eine DOS-Rettungsdiskette, mit der Sie den Rechner starten. Anschließend führen Sie unter DOS das Kommando `fdisk /mbr` aus.
- Bei Windows 2000/XP starten Sie den Rechner mit der Windows-Installations-CD und starten mit (R), (K) (Windows 2000) bzw. nur mit (R) (Windows XP) die so genannte Wiederherstellungskonsole. Dort können Sie aus einer Liste Ihre Windows-Installation auswählen. Nach der Eingabe Ihres Administrator-Passworts führen Sie das Kommando `FIXMBR` aus. Anschließend starten Sie den Computer mit `EXIT` neu.

2.10.2 X/KDE/Gnome startet nicht

Es kann vorkommen, dass Linux zwar gestartet wird, allerdings nur im Textmodus. Das X Window System (eventuell in Kombination mit KDE oder Gnome) wird dagegen nicht gestartet.

Dieses Verhalten ist nicht unbedingt ein Fehler, sondern kann durchaus beabsichtigt sein. Um X manuell zu starten, loggen Sie sich im Textmodus ein (geben Sie Ihren Benutzernamen und das Passwort ein) und führen dann das Kommando `startx` aus.

Wenn das klappt, sollten Sie (als Benutzer `root`) die Datei `/etc/inittab` ändern und dort die Zeile mit dem Schlüsselwort *initdefault* ändern. In dieser Zeile wird die Nummer oder der Buchstabe eines so genannten Runlevels angegeben. Bei den meisten Distributionen lautet die Nummer des Runlevels mit automatischem X-Start 5. Ab dem nächsten Rechnerstart wird X automatisch ausgeführt.

```
# in /etc/inittab
id:5:initdefault:
```

Sollte `startx` nicht zum Erfolg führen, resultieren die Probleme wahrscheinlich aus der falschen oder noch gar nicht erfolgten Konfiguration des X Window Systems.

2.10.3 Die Maus funktioniert nicht oder nur teilweise

Normalerweise sollten die Grundfunktionen der Maus – also deren Bewegung und die Maustasten – auf Anhieb funktionieren. Schon eher sind Probleme mit einem eventuell vorhandenen Mausrad zu erwarten, das eventuell ganz einfach ignoriert wird. In manchen Fällen ist in der X-Konfiguration ein Fehler (siehe 629) oder der falsche Treiber gewählt. Auch funktionieren manche USB-Mäuse (noch) nicht unter Linux.

2.10.4 Die Tastatur funktioniert nicht

Tastaturprobleme äußern sich im Regelfall dadurch, dass statt der gewünschten Buchstaben andere Zeichen erscheinen. Die Ursache ist fast immer eine falsche Einstellung des Tastatur-Layouts (siehe oben).

2.10.5 Menüs erscheinen in der falschen Sprache

Alle Linux-Programme sind in der Lage, Fehlermeldungen, Menüs etc. in englischer Sprache auszugeben. Sehr viele Programme (insbesondere KDE- und Gnome-Programme) stellen darüber hinaus aber auch Menüs in vielen Landessprachen zur Verfügung. Unter KDE können Sie die Sprache auch abweichend von den systemweiten Vorgaben einstellen.

2.11 Systemveränderungen, Erweiterungen und Updates

Wenn Ihr Linux-System einmal stabil läuft, wollen Sie es zumeist nach Ihren eigenen Vorstellungen konfigurieren, erweitern, aktualisieren etc. Informationen zu diesen Themen sind gleichsam über das gesamte Buch verstreut.

Je nach Distribution existieren verschiedene Kommandos und Programme, mit denen im laufenden Betrieb weitere Software-Pakete installiert, aktualisiert oder entfernt werden können. Sie sollten zur Installation generell nur das von der jeweiligen Distribution vorgesehene Tool verwenden und nur zur Distribution passende Pakete installieren. (Dieser Rat gilt insbesondere für Linux-Einsteiger!)

Zuerst müssen Sie sich fragen, was Sie eigentlich aktualisieren möchten: ein bestimmtes Software-Paket, den Kernel oder die ganze Distribution?

- **Update einzelner Programme:** Fast alle Distributionen bieten Werkzeuge an, mit denen sich Programm-Updates (vor allem zur Behebung von Sicherheitsproblemen) durch einen oder wenige Mausklicks erledigen lassen.
- **Update der gesamten Distribution:** Alle paar Monate erscheinen neue Versionen der gängigen Linux-Distributionen. Die meisten Distributionen bieten bei der Installation eine Update-Funktion an. Damit wird die neue Version gleichsam über die bisherige installiert. Dabei werden alte Pakete einfach durch neuere Versionen ersetzt. Die bisher durchgeführte Konfiguration bleibt aber weitestgehend erhalten.

  Was in der Theorie toll klingt, funktioniert in der Praxis leider ziemlich schlecht. Nach dem Update funktionieren meist diverse Programme nicht mehr wie vorher und die Suche nach den Fehlern kann zeitraubend sein. Falls Sie dennoch ein Distributions-Update versuchen möchten, sollten Sie auf jeden Fall Sicherheitskopien von allen Konfigurationsdateien erstellen (am

besten kopieren Sie den Inhalt des gesamten `/etc`-Verzeichnisses an einen sicheren Ort). Ebenso selbstverständlich sollte ein Backup Ihrer persönlichen Daten sein (und das nicht nur bei einem Distributions-Update, sondern regelmäßig)!

Tipp

Wenn Sie einmal ein stabil laufendes Linux-System haben, gibt es zumeist nur wenige Gründe, ein komplettes Update durchzuführen! Lassen Sie sich nicht von der allgemeinen Versionshysterie anstecken! Für den stabilen Betrieb ist es meist vollkommen ausreichend, regelmäßig alle Sicherheits-Updates durchzuführen. Viele Linux-Server im kommerziellen Einsatz laufen auf diese Art und Weise oft jahrelang!

2.12 Linux wieder entfernen

Falls Sie zu einem späteren Zeitpunkt den von Linux beanspruchten Platz auf der Festplatte wieder freigeben möchten, müssen Sie Linux deinstallieren. Dazu sind zumeist zwei Schritte erforderlich:

- Sie müssen den Boot-Loader entfernen und stattdessen den Urzustand des Bootsektors der Festplatte wiederherstellen. Die Vorgehensweise für mehrere Windows-Versionen wurde auf Seite 66 beschrieben.
- Sie müssen alle Linux-Partitionen löschen, damit Sie den Platz später wieder unter Windows nutzen können. Diesen Schritt können Sie grundsätzlich unter Windows durchführen. Noch besser ist es aber, dafür Knoppix oder das Rescue-System Ihrer Distribution einzusetzen – beides kann direkt von einer CD gestartet werden. (Generell gilt die Regel, dass Partitionen eines bestimmten Betriebssystem nur mit den Werkzeugen dieses Betriebssystems verändert werden sollten. Daher sollten Linux-Partitionen unter Linux gelöscht werden. Allerdings ist es unmöglich, die Systempartition eines laufenden Linux-Systems zu löschen. Deswegen ist eine Linux-Variante erforderlich, die direkt von der CD ausgeführt wird.)

  Zum eigentlichen Löschen der Linux-Distributionen müssen Sie zumeist das Programm `fdisk` verwenden. Die komfortableren Partitionierungswerkzeuge, die während der Installation zur Verfügung stehen, sind bei den meisten Distributionen im Rescue-System leider nicht enthalten. Falls Sie eine Knoppix-CD besitzen, können Sie statt `fdisk` das Programm `qtparted` einsetzen.

Kapitel 3

Linux-Einstieg

Einführung

Dieses Kapitel hilft, die ersten Schritte unter Linux durchzuführen: einloggen, ausloggen, Rechner herunterfahren, grundlegende Kommandos in einer Textkonsole ausführen, eine Textdatei editieren etc. Das Kapitel vermittelt auch ein minimales Grundlagenwissen über die Datei-, Prozess- und Benutzerverwaltung von Linux. Ausserdem verraten wir Ihnen, wo Sie im installierten System bzw. im Internet nach Online-Dokumentationen suchen können.

Ein Grundproblem bei einer allgemeinen Beschreibung von Linux besteht darin, dass fast jedes Detail von Linux frei konfiguriert werden kann. Daher kann es sein, dass ein bestimmtes Tastenkürzel unter Red Hat eine andere Reaktion hervorruft als unter Suse. Aus diesem Grund gibt es in diesem Kapitel viele Formulierungen mit *meistens*, *gewöhnlich* etc. Das ist leider nicht zu ändern. Die einzige Alternative bestünde darin, Linux so zu beschreiben, wie es bei einer ganz bestimmten Distribution (und da wiederum bei einer ganz bestimmten Versionsnummer) funktioniert.

Linux-Benutzer sind „mündig“! Was heißt das? Dazu ein Beispiel: Wenn Sie ein Kommando eingeben, das die gesamte Platte löscht, fragt das Lösch-Programm nicht noch einmal nach, ob Sie das auch wirklich wollen, sondern löscht die Platte sofort.

Manche Themen werden in diesem Kapitel nur ganz kurz angeschnitten, denn um alle Eingenschaften und Möglichkeiten, Höhen und Tiefen von Linux auszubreiten, reicht ein Buch nicht. Manches müssen Sie auch gar nicht wissen (das gilt auch für andere Betriebssysteme) und manches wollen Sie vielleicht zu einem ganz bestimmten Zeitpunkt in Erfahrung bringen – das ist übrigens das Schöne an Linux: Man kann erst einmal relativ schlicht anfangen und sich erst dann in Details vertiefen wenn es nötig ist. So werden Sie anfangen, die Skriptsprache

der Shell zu lernen, wenn Sie dieselben fünf Kommandos mehrmals eingegeben und sich dabei 17-mal vertippt haben.

Relativ bald beschäftigen müssen Sie sich mit den Multiuser-Eigenschaften von Linux. Jedes Objekt (Datei, Gerät etc.) hat bei Linux einen Eigentümer und Zugriffsrechte, die regeln, inwieweit andere Benutzer darauf zugreifen dürfen. Bei Linux kommen Sie daher auch um etwas Benutzerverwaltung nicht herum. Schon bei der Installation wurden Sie ja aufgefordert, sich selbst als Benutzer einzurichten. Daneben gibt es noch mindestens einen anderen Benutzer: den Systemverwalter (meist `root` genannt).

3.1 Linux starten und beenden

Um Linux zu starten, müssen Sie Ihren Rechner neu starten. Je nachdem, wie Sie Linux installiert haben, erfolgt der Start von Linux durch eine Boot-Diskette oder direkt von der Festplatte. Unabhängig davon wird hierfür ein Boot-Loader (meistens GRUB) eingesetzt. Das Programm macht sich zumeist durch ein Menü bemerkbar, in dem Sie zwischen Windows und Linux auswählen können. Der Linux-Boot-Prozess dauert etwa 15-60 Sekunden, je nachdem, wie schnell Ihre Hardware ist. Bei vielen Distributionen erscheint während dieser Zeit ein Fortschrittsbalken oder Splash-Screen. Andere Distributionen zeigen hingegen viele Detailinformationen über den Systemstart an. Diese Informationen sind aber nur dann von Interesse, wenn etwas nicht funktionieren sollte. Sie lassen sich übrigens mit dem Kommando `dmesg` jederzeit wieder anzeigen.

Login

Im Normalfall wird am Ende des Bootvorgangs das Grafiksystem (X Window System) gestartet. Darin erscheint eine Login-Box, in der Sie sich mit Ihrem Benutzernamen und dem Passwort anmelden müssen.

Nach etwa 20 bis 30 Sekunden sollte eine so genannte Login-Aufforderung erscheinen. Anders als bei (alten) Windows-Versionen müssen Sie sich immer mit Name und Passwort anmelden, bevor Sie mit Linux arbeiten können. Das ist deswegen notwendig, weil Linux zwischen verschiedenen Benutzern mit unterschiedlichen Privilegien unterscheidet.

Auf jedem Linux-Rechner gibt es zumindest den Benutzer „root“, der als Systemadministrator gilt und unbeschränkte Rechte hat. Sein Passwort haben Sie während der Installation angegeben. Wenn nicht, sollten Sie das so bald wie möglich nachholen. Im Regelfall haben Sie während der Konfiguration auch schon einen oder mehrere gewöhnliche Benutzer definiert.

Hinweis

Mit `root` gelten Sie als Systemadministrator. Damit haben Sie Zugriff auf alle Dateien, dürfen uneingeschränkt alle Programme ausführen und können (durchaus unbeabsichtigt) eine ganze Menge Schaden anrichten. Arbeiten Sie nur dann als `root`, wenn es wirklich erforderlich ist! Nur während der Konfiguration des Systems ist es zweckmäßig, als `root` zu arbeiten. (Gewöhnliche Benutzer dürfen globale Konfigurationsdateien nicht ändern.)

Desktop auswählen: Im Gegensatz zu Windows ist die grafische Benutzerschnittstelle nicht integraler Bestandteil des Betriebssystems, sondern eine weitere Schicht darüber. Deshalb haben Sie auch die Wahl zwischen unterschiedlichen „window managers“ und „desktop environments“, wobei bei Linux eigentlich nur zwei anzutreffen sind, **KDE** oder **Gnome** (obwohl Sie auch hier wieder die Wahl haben, es ganz anders zu machen). Bei vielen Distributionen werden automatisch sowohl KDE als auch Gnome installiert. In diesem Fall können Sie während des Logins auswählen, welches Desktop-System gestartet werden soll.

Wie diese Auswahl erfolgt, hängt ebenfalls von der Distribution ab: Entweder gibt es in der Login-Box ein Listenfeld, in dem Sie den gewünschten Desktop auswählen können (z. B. bei Suse); oder es gibt am oberen oder unteren Bildschirmrand ein Menü zur Auswahl des Desktops (z. B. bei Red Hat).

Automatischer Login: Linux kann so konfiguriert werden, dass nach dem Rechnerstart ein automatischer Login erfolgt. Das ist zwar bequem, aber aus Sicherheitsgründen nicht empfehlenswert. Wollen Sie sich unter einem anderen Namen einloggen, müssen Sie sich vorher ausloggen.

Login im Textmodus: Linux kann so konfiguriert werden, dass der Login im Textmodus erfolgt. In diesem Fall führen Sie nach dem Login `startx` aus, um das X Window System und damit KDE oder Gnome zu starten. (Ob Linux im Text- oder Grafikmodus startet, hängt vom Inhalt der Datei `/etc/inittab` ab.

Falls `startx` zu Fehlermeldungen führt, hat die Konfiguration des X Window Systems während der Installation nicht korrekt funktioniert. Eine Anleitung, wie Sie diesen Schritt selbst erledigen können, finden Sie in Kapitel 13.

Logout

Mit dem KDE- bzw. Gnome-Menüeintrag *Abmelden* führen Sie einen Logout durch. Dabei werden sämtliche am Desktop laufenden Programme beendet. (Sichern Sie vorher alle noch offenen Dateien!) Der Logout führt zurück zur Login-Box, in der Sie sich nun neu einloggen oder den Rechner herunterfahren können.

Falls Sie im Textmodus arbeiten, drücken Sie zum Logout einfach Strg+D oder führen das Kommando `exit` aus.

Tipp

Das ASCII-Zeichen `Strg-D` ist bei Linux und UNIX das Dateiende-Zeichen. Sie können bei fast jedem Programm von Linux die Eingabe durch die Tastenkombination Strg+D beenden. Beim Login im Textmodus arbeiten Sie in der Regel mit der Shell, die wie jedes andere Programm behandelt wird und die Eingabe des Dateiendezeichens sagt der Shell „Ich bin fertig, hör auf!“. Das macht sie dann auch und es erscheint wieder der Login-Prompt.

Linux beenden (Shutdown)

Mindestens ebenso wichtig wie das korrekte Einloggen ist ein ordnungsgemäßes Beenden des Systems. Auf keinen Fall dürfen Sie den Rechner einfach ausschalten! Sie riskieren damit im ungünstigsten Fall ein zerstörtes Dateisystem. Das ist übrigens bei Windows genauso. Das Verlassen von Linux sieht unterschiedlich aus, je nachdem, ob der Login unter X oder im Textmodus erfolgte.

- **Abmelden bei grafischem Login:** Je nach Konfiguration können Sie bereits beim Ausloggen oder in der danach erscheinenden Login-Box den Rechner herunterfahren (Button *Beenden* oder *Neu starten*, je nachdem, ob Sie den Rechner ausschalten möchten oder nicht).

- **Abmelden im Textmodus:** Im Textmodus erfolgt ein ordnungsgemäßes Herunterfahren des Systems mit dem Kommando `shutdown -h now` (Kurzform: `halt`). Diese Kommandos dürfen allerdings nur von `root` ausgeführt werden.

  Auf vielen Linux-Systemen gibt es eine bequemere Alternative zum `shutdown`-Kommando: Drücken Sie im Textmodus einfach die Tasten Strg+Alt+Entf. Damit kann jeder Anwender (nicht nur `root`) das Kommando `shutdown` ausführen. (Die Reaktion auf die Tastenkombination Strg + Alt + Entf wird durch die Datei `/etc/initab` gesteuert – siehe Seite 607.)

  Strg+Alt+Entf bewirkt nur im Textmodus ein Herunterfahren, nicht aber unter X. Falls Sie unter X arbeiten, müssen Sie vorher mit Strg+Alt+F1 in eine Textkonsole wechseln.

- **Reboot im Textmodus:** Im Textmodus erfolgt ein ordnungsgemäßes Neustarten des Systems mit dem Kommando `shutdown -r now` (Kurzform: `reboot`). Diese Kommandos dürfen allerdings nur von `root` ausgeführt werden.

Tipp

`shutdown` kann nur von `root` ausgeführt werden. Wenn Linux auf den „Affengriff" (Strg + Alt + Entf) nicht reagiert (je nach Konfiguration in der Datei `/etc/inittab`) und Sie kein `root`-Passwort besitzen, den Rechner aber dennoch neu starten müssen, sollten Sie vorher zumindest das Kommando `sync` ausführen. Damit werden alle gepufferten Schreibzugriffe auf die Festplatte ausgeführt. Unmittelbar anschließend schalten Sie Ihren Rechner aus. Das ist allerdings nur eine Notlösung zur Schadensminimierung.

3.2 Grafische Benutzeroberflächen (KDE und Gnome)

Wie bereits erwähnt, können Sie unter Linux entweder im Grafikmodus (X Window System) oder mit Textkonsolen arbeiten. Das X Window System stellt allerdings nur die Basis für darauf aufbauende Benutzeroberflächen wie KDE oder Gnome dar. Wenn Sie also KDE oder Gnome verwenden, läuft im Hintergrund immer auch X. Daher gelten alle Textpassagen dieses Buchs, die sich auf X beziehen, automatisch auch für KDE oder Gnome!

Grundsätzlich können Sie X aber auch ohne KDE und Gnome nutzen. Dazu muss beim Start von X bzw. beim Einloggen ein so genannter „Window Manager" gestartet werden. Der Vorteil gegenüber KDE und Gnome besteht darin, dass die meisten Window Manager viel geringere Hardware-Anforderungen stellen. Allerdings ist auch der Bedienungskomfort geringer.

Falls Sie bereits eingeloggt sind, werden Sie sich vielleicht fragen, welches Desktop-System nun eigentlich gerade läuft. Die Antwort darauf ist gar nicht so einfach, weil das Aussehen von KDE und Gnome bei vielen Distributionen stark verändert und mit eigenen Logos versehen wird. Besonders weit gehen in dieser Hinsicht Red Hat bzw. Fedora, deren BlueCurve-Konfiguration KDE und Gnome beinahe identisch aussehen lässt. Ein einfaches Erkennungskriterium ist das Panel: Mit der rechten Maustaste gelangen Sie in ein Kontextmenü, in dem Sie bei KDE den Menüeintrag *Hilfe* → *Über KDE* finden, bei Gnome *Info zu Gnome*.

3.2.1 KDE-Schnelleinstieg

Prinzipiell orientiert sich die Bedienung von KDE stark an der von Windows. In KDE gibt es eine so genannte Kontrollleiste (normalerweise am unteren Bildschirmrand), über die Sie Programme starten sowie KDE verlassen können. Darüber hinaus gibt es eine Programmleiste, in der für jedes laufende Programm ein Button (Icon) angezeigt wird. Die Icons sind in die Programmleiste integriert und ermöglichen wie die Task-Leiste unter Windows den raschen Wechsel zwischen den Programmen.

Im Unterschied zu Windows kennt KDE auch den Begriff von Arbeitsflächen (die oft auch als virtuelle Desktops bezeichnet werden). Normalerweise sind meist vier Desktops – gewissermaßen vier verschiedene Bildschirme – vorgesehen, zwischen denen Sie auf Knopfdruck wechseln können. Der Sinn dieser Arbeitsflächen besteht darin, dass Sie unterschiedliche Programme in unterschiedlichen Arbeitsflächen starten können. Damit vermeiden Sie das sonst übliche Chaos sich überlagernder Fenster.

Die mittlere und die rechte Maustaste haben auf der Arbeitsoberfläche eine besondere Bedeutung. Die mittlere Taste führt zu einem Menü, in dem alle zurzeit laufenden Programme aufgezählt sind (wie (Alt) + (Tab) unter Windows). Das Menü bietet also eine einfache Möglichkeit, zwischen Programmen zu wechseln. Die rechte Taste führt in ein Menü mit einigen elementaren KDE-Kommandos (Online-Hilfe, KDE verlassen etc.).

Auf den Arbeitsflächen werden neben Fenstern auch Icons dargestellt. Diese haben dieselbe Bedeutung wie unter Windows. Sie ermöglichen den raschen Start von Programmen bzw. den bequemen Wechsel in ein bestimmtes Verzeichnis bzw. den einfachen Zugriff auf die Verzeichnisse einer CD-ROM, Diskette oder Windows-Partition.

Zu den wichtigsten KDE-Programmen zählt der Konqueror. Dabei handelt es sich um ein Universalprogramm, das gleichermaßen Webbrowser und Datei-Manager ist. Das Programm kann durch einen Klick auf das Icon *Persönliches Verzeichnis* im KDE-Menü gestartet werden.

Die Konfiguration der meisten KDE-Komponenten (Aussehen der Fenster, Platzierung der Kontrollleiste, Bildschirmhintergrund, Bildschirmschoner etc.) erfolgt durch das so genannte Kontrollzentrum. Auch dieses Programm wird über das KDE-Startmenü aufgerufen.

3.2.2 Gnome-Schnelleinstieg

Ähnlich wie bei KDE gibt es auch bei der Bedienung von Gnome viele Ähnlichkeiten zu Windows. Auch Gnome wird in erster Linie durch eine Kontrollleiste gesteuert. Sie enthält das Startmenü, einige Icons zum Schnellstart wichtiger Programme sowie Icons für alle zurzeit laufenden Programme. Wie bei KDE können die Fenster auf mehrere Arbeitsflächen (Desktops) verteilt werden.

Bei den Icons in der Kontrollleiste reicht ein einfacher Mausklick zum Start bzw. zur Aktivierung. Bei Icons auf dem Desktop ist dagegen ein Doppelklick erforderlich.

Die mittlere und die rechte Maustaste haben auf der Arbeitsoberfläche eine besondere Bedeutung. Mit der mittleren Taste können Sie zu einem der laufenden Programme wechseln, die aktive Arbeitsfläche auswählen, Programme starten etc. Mit der rechten Taste sind nur einige ausgewählte Operationen möglich.

Zu den wichtigsten Gnome-Programmen zählt der Datei-Manager Nautilus. Das Programm kann über das Start- oder Orte-Menü gestartet werden.

Die Konfiguration von Gnome (Aussehen der Fenster, Platzierung der Kontrollleiste, Bildschirmhintergrund, Bildschirmschoner etc.) erfolgt über das Menü *System → Einstellungen*.

3.3 Tastatur, Maus und Zwischenablage

3.3.1 Wichtige Tastenkürzel

Welche Tastenkürzel zur Verfügung stehen, hängt davon ab, ob Sie im Grafikmodus oder in einer Textkonsole arbeiten. Dieser Abschnitt behandelt den Grafikmodus, Tipps zur Verwendung von Textkonsolen folgen im nächsten Abschnitt. Im Grafikmodus können Tastenkürzel durch drei Programmebenen definiert werden:

- Das X Window System ist für die elementaren Funktionen des Grafiksystems verantwortlich. Es stellt nur relativ wenige Tastenkürzel zur Verfügung.
- Die Desktop-Systeme KDE und Gnome definieren weitere Tastenkürzel.
- Schließlich hängen die verfügbaren Tastenkürzel vom gerade laufenden Programm ab. Je nachdem, ob Sie mit Mozilla surfen, mit OpenOffice einen Brief oder im Editor vi ein Programm schreiben – in jedem Fall gelten andere Tastenkürzel, die hier natürlich nicht beschrieben werden. Es gibt aber auch bei Linux Konventionen, an die sich viele Programme halten.

Tastenkürzel unter X	
Strg+Alt + Backspace	X-Server gewaltsam beenden
Strg+Alt+F1 bis F6	In Textkonsole 1 bis 6 wechseln
Alt+F7	Zurück zu X wechseln
Alt+F5	Zurück zu X wechseln (bei Knoppix)
Strg+Alt++ / -	Wechsel des XFree86-Grafikmodus (falls mehrere Modi definiert sind)

Die wichtigsten KDE-Tastenkürzel

Alt+Tab	Aktives Programm wechseln
Strg+Tab	Virtuellen Desktop wechseln
Strg+F1 bis F12	Virtuellen Desktop 1 bis 12 aktivieren
Alt+F1	KDE-Menü aufrufen
Alt+F2	Programm starten
Alt+F3	Fenstermenü anzeigen
Alt+F4	Fenster schließen, Programm beenden
Alt+F5	Liste aller Fenster anzeigen
Strg+Alt+Esc	Programm gewaltsam beenden (`xkill`)
Strg+Alt+Entf	KDE beenden

Die wichtigsten Gnome-Tastenkürzel	
Alt+Tab	Aktives Programm wechseln
Strg+Alt+Tab	Focus zwischen Desktop und Panel wechseln
Alt+F1	Gnome-Menü aufrufen
Alt+F2	Programm ausführen
Alt+F4	Fenster schließen, Programm beenden
Strg+Alt+←	Voriger virtueller Desktop
Strg+Alt+→	Nächster virtueller Desktop

Immer mehr Programme, insbesondere solche für KDE und Gnome, halten sich bei vielen Tastenkürzeln an die unter Windows üblichen Konventionen. Das betrifft beispielsweise das Kopieren von Text in die Zwischenablage mit Strg+Einf oder Strg+C, das Einfügen des kopierten Texts mit Shift+Einf oder Strg+V oder das Speichern einer Datei mit Strg+S.

Daneben gibt es aber auch viele, zumeist ältere Programme, in denen die früher unter Unix üblichen Konventionen gelten. Das betrifft insbesondere fast alle Programme, die im Textmodus oder in einem Shell-Fenster laufen. Die folgende Tabelle fasst die wichtigsten derartigen Tastenkürzel zusammen.

Typische Linux-Tastenkürzel (je nach Anwendung)	
Strg+A	Cursor an den Zeilenanfang (wie Pos1)
Strg+C	Programm abbrechen
Strg+E	Cursor an das Ende der Zeile (wie End)
Strg+D	Ein Zeichen löschen (wie Entf)
Strg+H	Zeichen rückwärts löschen (wie Backspace)
Strg+K	Zeile ab Cursor löschen
Strg+Z	Programm unterbrechen (Fortsetzung mit `fg`)
Tab	Datei- und Kommandonamen vervollständigen

Die Kommandoerweiterung mit Tab gilt nur bei der Shell. Wenn Sie ein Kommando ausführen oder einen Dateinamen angeben möchten, brauchen Sie nur die

ersten Buchstaben eintippen. Anschließend drücken Sie Tab. Wenn der Dateiname bereits eindeutig erkennbar ist, wird er vollständig ergänzt, sonst nur so weit, bis sich mehrere Möglichkeiten ergeben. Ein zweimaliges Drücken von Tab bewirkt, dass eine Liste aller Dateinamen angezeigt wird, die mit den bereits eingegebenen Anfangsbuchstaben beginnen.

3.3.2 Eingabe ausländischer Sonderzeichen

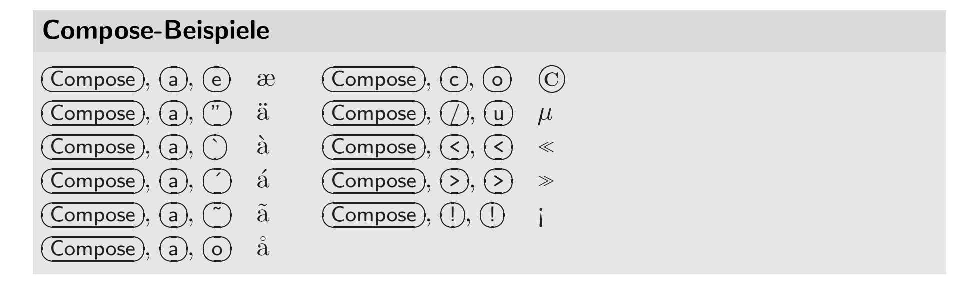
Sonderzeichen zusammensetzen: Es gibt unter Linux zwei Möglichkeiten, Sonderzeichen zusammenzusetzen:

- Mit einer Compose-Taste können Sie die zwei folgenden Zeichen gleichsam vereinen. Beispielsweise liefert Compose, A, E das Zeichen Æ.
- Je nach Konfiguration werden die Zeichen ˜ ˆ ' ' automatisch mit dem nächsten Zeichen kombiniert. ˜, a ergibt daher das Zeichen ã. (Dieser Eingabemodus kann allerdings aufgrund der Option `nodeadkeys` in `/etc/X11/XF86Config` deaktiviert sein.)

Welche Taste(nkombination) als Compose-Taste gilt, hängt von der Distribution ab. Die Compose-Taste wird in der X-Konfiguration festgelegt und kann auch dort geändert werden. Beachten Sie, dass Sie Shift immer vor der jeweils zweiten Taste drücken müssen.

Debian, Knoppix:	Shift+AltGr
Mandriva:	Shift+AltGr
Red Hat, Fedora:	Shift+AltGr
Suse:	Shift+AltGr
	Shift+Rechte Strg-Taste
	Shift+Rechte Windows-Taste

Welche Sonderzeichen wie zusammengefügt werden können, hängt vom aktiven Zeichensatz ab. Die folgenden Beispiele gelten für den ISO-Latin-1-Zeichensatz, der in vielen Ländern Westeuropas gebräuchlich ist:



Compose-Beispiele			
Compose, a, e	æ	Compose, c, o	©
Compose, a, "	ä	Compose, /, u	µ
Compose, a, `	à	Compose, <, <	≪
Compose, a, ´	á	Compose, >, >	≫
Compose, a, ˜	ã	Compose, !, !	¡
Compose, a, o	å		

Euro und Cent: Bei fast allen Distributionen ist die Tastatur so konfiguriert, dass das Tastenkürzel (AltGr)+(E) der X-Keysym-Zeichenkette `EuroSign` zugeordnet ist. Ebenso ist (AltGr)+(C) dem Cent-Symbol zugeordnet. (Diese Tastenkürzel gelten für den deutschen Sprachraum. In anderen Ländern sind zum Teil andere Tastenkürzel zur Eingabe dieser Symbole üblich.) Allerdings kommen manche ältere Programme mit dieser Eingabe nicht zurecht. KDE- oder Gnome-Programme sollten aber keine Probleme verursachen.

3.3.3 Verwendung der Maus

Einfach- oder Doppelklick: Unter Gnome ist wie unter Windows für viele Operationen – etwa das Öffnen einer Datei – ein Doppelklick erforderlich. In KDE werden dagegen viele Mausoperationen wie früher schon bei X durch einen einfachen Mausklick ausgeführt. KDE kann allerdings ebenfalls im Doppelklickmodus konfiguriert werden.

Texte mit der Maus kopieren und einfügen: In fast allen X-Programmen können Sie die Maus dazu verwenden, Textausschnitte zu kopieren und diese anschließend an einer anderen Stelle (oder in einem anderen Programm) wieder einzufügen. Zum Markieren von Textausschnitten bewegen Sie die Maus einfach mit gedrückter linker Maustaste über den Text. Der so markierte Text wird dabei automatisch in einen Puffer kopiert. Sobald Sie die mittlere Maustaste drücken, wird der Text dort eingefügt, wo der aktive Eingabecursor steht.

Das Markieren und Kopieren erfolgt also allein mit der Maus, ohne Tastatur. Wenn Sie sich einmal an diese Methode gewöhnt haben, werden Sie sich immer fragen, warum das unter Windows nicht möglich ist. Besitzer von Mäusen mit nur zwei Tasten sind nicht ausgeschlossen. Diese sind in der Regel so eingerichtet, dass man durch gleichzeitiges Drücken beider Tasten die mittlere Taste „simulieren“ kann.

Bedienung von Bildlaufleisten: Bei allen modernen Programmen kann die Bildlaufleiste mit der linken Maustaste verschoben werden. Es gibt allerdings auch noch einige alte X-Programme (beispielsweise `xterm` oder `ghostview`), bei denen die Bildlaufleiste nur mit der mittleren Maustaste verschoben werden kann. Bei derartigen Programmen bewirkt das Drücken der linken Maustaste, dass der Schiebebalken nach unten bewegt wird. Das Drücken der rechten Maustaste bewegt den Schiebebalken nach oben. Dabei gibt die aktuelle Position der Maus an, wie weit der Schiebebalken bewegt wird: Wenn die Maus im Schiebebalken ganz oben steht, wird das Schiebefeld nur ein wenig bewegt (eine Zeile). Wenn die Maus dagegen ganz unten steht, wird das Schiebefeld eine ganze Seite weiterbewegt.

Eingabefokus: Bei einigen älteren X-Programmen (nicht bei KDE- oder Gnome-Programmen) kann bei Dialogen nur dann Text in Eingabefelder eingegeben werden, wenn sich die Maus über diesem Feld befindet. Der Eingabefokus hängt also nicht nur davon ab, welches Programm gerade aktiv ist, sondern auch davon, wo sich die Maus befindet.

3.3.4 Maussteuerung per Tastatur

Wenn die Maus nicht funktioniert, können Sie den Mauszeiger zur Not auch mit der Tastatur steuern. Dazu müssen Sie mit (Shift)+(Strg)+(Num) einen speziellen Tastaturmodus aktivieren. Für die Tasten des Ziffernblocks der Tastatur gilt nun folgende Zuordnung:

Tastenkürzel zur Maussteuerung	
(4), (6)	Maus nach links bzw. rechts bewegen
(2), (8)	Maus nach unten bzw. oben bewegen
(5)	Linke Maustaste kurz drücken
(+)	Doppelklick
(0)	Maustaste bleibend drücken ((5) löst die Taste wieder)
(-)	Auf die rechte Maustaste umschalten ((5), (+) und (0) gelten jetzt für die rechte Maustaste)
(*)	Wieder auf die linke Maustaste umschalten

Wenn Sie den numerischen Tastenblock wieder normal nutzen möchten, schalten Sie die Maussteuerung durch nochmaliges Drücken von (Shift)+(Strg)+(Num) wieder aus.

3.3.5 Zwischenablage

Wie gerade erwähnt wurde, gilt jeder mit der Maus markierte Text als eine Art Ad-hoc-Zwischenablage. Solange die Markierung besteht, kann der markierte Text mit der mittleren Maustaste in ein anderes Programm eingefügt werden. Der Vorteil dieses Verfahrens besteht darin, dass es ohne Tastatur funktioniert. Das Verfahren hat aber auch Nachteile: Durch jede neue Markierung wird die bisherige Markierung (und damit die Ad-hoc-Zwischenablage) gelöscht, was oft lästig ist.

Deswegen bieten viele Programme (alle KDE- und Gnome-Programme, Mozilla etc.) darüber hinaus die Möglichkeit, wie unter Windows mit einem bestimmten Tastenkürzel Texte in eine eigene Zwischenablage zu kopieren. Das Tastenkürzel lautet meist wie unter Windows (Strg)+(C) bzw. (Strg)+(Einfg). Die Tastenkürzel zum Einfügen des Inhalts der Zwischenablage lauten üblicherweise (Strg)+(V) bzw. (Shift)+(Einfg).

Der Inhalt der Zwischenablage kann mit dem Programm `xclipboard` angesehen werden. Das Programm gibt auch Zugriff auf ältere Inhalte der Zwischenablage. Unter KDE wird die Zwischenablage noch komfortabler durch das Hintergrundprogramm `klipper` verwaltet. Dieses Programm wird über ein Notizblocksymbol in der Taskleiste bedient.

3.4 Textkonsolen und Shell-Fenster

In den letzten Jahren sind die Desktop-Systeme KDE und Gnome so populär geworden, dass viele Linux-Einsteiger beinahe ausschließlich mit Programmen arbeiten, die über das Startmenü zugänglich sind (manche Distributionen verhalten sich auch schon genauso resistent gegen Änderungen durch den Benutzer wie Windows). Diese Programme sind aber nur ein kleiner Teil des Linux/Unix-Universums.

Der andere Teil sind textorientierte Kommandos, die in einer Textkonsole oder in einem Shell-Fenster ausgeführt werden. Dazu zählen beispielsweise Kommandos zur Verwaltung des Dateisystems (`ls`, `cp`, `mv`, `ln`, `rm` etc.), Suchkommandos (`find`, `grep`, `locate`), Programme zur Prozessverwaltung (`crontab`, `kill`, `nohup`, `ps`) und Netzwerk-Tools (`ping`, `ifconfig`, `traceroute`, `ssh`, `scp`, `wget`). Jedes dieser Kommandos hat eine einzige, genau definierte Aufgabe. Durch Kombination mehrerer Kommandos können Sie ganz schnell individuelle und mächtige Werkzeuge basteln.

Als Linux-Einsteiger können Sie sicher eine Weile ohne diese Kommandos leben. Fortgeschrittene Anwender müssen aber lernen, damit umzugehen. Deswegen bietet dieser Abschnitt einen ersten Einblick in das kommando-orientierte Arbeiten unter Linux.

3.4.1 Textkonsolen

Microsoft Windows können Sie ausschließlich im Grafikmodus verwenden. Für Linux gilt das nicht. Linux funktioniert sowohl im bereits beschriebenen Grafikmodus als auch in so genannten Textkonsolen. Natürlich ist das Arbeiten unter X bequemer und intuitiver. Mit Textkonsolen kommen Sie meist nur in Kontakt, wenn die Konfiguration von X noch nicht durchgeführt wurde bzw. nicht funktioniert hat. Auch Linux-Rechner, die als reine Server konfiguriert werden, laufen oft im Textmodus. Oft hat man auch ein Kommando schneller eingetippt, als wenn man mit der Maus endlos rumklickt. Textkonsolen haben auch im Zeitalter grafischer Benutzeroberflächen ihre Berechtigung und Sie sollten damit umgehen können. In der Standardeinstellung stehen meist sechs Textkonsolen zur Verfügung. Der Wechsel zwischen diesen Textkonsolen erfolgt mit (Alt)+(F1) für die erste Konsole, (Alt)+(F2) für die zweite usw.

Auch bei der Textkonsole müssen Sie sich einloggen. Wenn Sie mit der Arbeit fertig sind oder wenn Sie sich unter einem anderen Namen anmelden möchten, müssen Sie sich wieder ausloggen. Dazu drücken Sie einfach (Strg)+(D).

Linux ist ein Multitasking-System. Das bedeutet, dass Sie in der einen Konsole ein Kommando starten und bis dieses beendet ist, können Sie in der zweiten Konsole etwas anderes erledigen. Sie können sich auch in einer Konsole als `root` anmelden, um administrative Aufgaben zu erledigen, während Sie in der anderen Konsole unter Ihrem normalen Login-Namen E-Mails lesen.

Tastenkürzel in Textkonsolen	
Strg+Alt+Entf	Linux beenden (`shutdown`)
Alt+F1 bis F6	In Textkonsole 1 bis 6 wechseln
Alt+F7	Zu X wechseln
Alt+F5	Zu X wechseln (bei Knoppix)
Alt+→	In die vorige Textkonsole wechseln
Alt+←	In die nächste Textkonsole wechseln
Tab	Datei- und Kommandonamen vervollständigen
Shift+Bild ↑/↓	Vorwärts/rückwärts blättern
↑ / ↓	Durch die bisher ausgeführten Kommandos blättern

Mit Shift+Bild ↑ und Shift+Bild ↓ können Sie den Bildschirminhalt einer Textkonsole auf- und abscrollen. Auf diese Weise können Sie die Ergebnisse der zuletzt ausgeführten Programme nochmals ansehen, auch wenn sie bereits nach oben aus dem sichtbaren Bildschirmbereich hinausgeschoben wurden.

Beachten Sie bitte, dass die Verwendung von Sonderzeichen (also aller Zeichen außerhalb des 7-Bit-ASCII-Zeichensatzes) in der Konsole problematisch ist. Die Eingabe von Sonderzeichen funktioniert nur dann zufriedenstellend, wenn ein 8-Bit-Zeichensatz (z. B. Latin-1) und ein dazu passender Font aktiv sind. Die Konsole eignet sich hingegen nicht zur Eingabe und Darstellung von Unicode-Texten.

3.4.2 Konsolenfenster (Shell-Fenster)

Natürlich müssen Sie nicht in eine Textkonsole wechseln, um Kommandos auszuführen. Wenn Sie ein Desktop-System nutzen, öffnen Sie darin einfach ein Konsolenfenster. (Derartige Fenster werden auch Shell-Fenster oder (X-)Terminal-Fenster genannt.) Unter Suse/KDE lautet der Menüeintrag beispielsweise *System* → *Terminals* → *Konsole*. Unter Gnome kann ein Konsolenfenster noch bequemer einfach per Kontextmenü auf dem Desktop geöffnet werden. Falls Sie unter X arbeiten, aber weder KDE noch Gnome nutzen, starten Sie das Programm `xterm`.

Innerhalb von Konsolenfenstern können Sie wie in einer Textkonsole arbeiten. Der einzige Unterschied besteht darin, dass Sie dank einer Bildlaufleiste bequemer durch die bisherigen Ausgaben scrollen können.

Böse Zungen behaupten, X sei nur erfunden worden, damit Administratoren mehrere Konsolenfenster auf einem Bildschirm offen haben können.

3.4.3 Kommandos ausführen

Mit Linux werden zahllose Kommandos und Programme mitgeliefert. (Die Unterscheidung zwischen Kommandos und Programmen ist relativ willkürlich; es

handelt sich in jedem Fall um Programme. Als „Kommandos“ werden zumeist kleinere Programme bezeichnet, die zum alltäglichen Umgang mit Linux erforderlich sind.) Zum Ausführen von Kommandos geben Sie in der Textkonsole oder im Shell-Fenster einfach den Kommandonamen, eventuell einige Parameter und schließlich ⏎ ein. Wer früher noch nit DOS gearbeitet hat, kennt das ja. In diesem Buch werden die Eingabe eines Kommandos (fett) und das Ergebnis so dargestellt:

```
user$  ls -l       # Inhalt des aktuellen Verzeichnisses
lrwxrwxrwx   1 user users    14 Nov 16 13:08 linux -> /usr/src/linux/
-rw-------   1 user users  7081 Nov 21 15:50 test1
-rw-r--r--   1 user users    46 Nov 21 09:29 test2
```

Die Zeichenfolge `user$` in der ersten Zeile ist ein Prompt des Betriebssystems (den man beliebig ändern kann). Das $-Zeichen bedeutet, dass man als gewöhnlicher Benutzer eingeloggt ist und das Kommando ohne besondere Privilegien ausgeführt wird. Wenn statt des „$“ ein „#“ angegeben ist, wird das Kommando hingegen von dem Superuser `root` ausgeführt. Den Eingabe-Prompt dürfen Sie natürlich *nicht* mit eingeben!

Die Optionen eines Kommandos (wie oben das „-l“) werden mit einem „-“ eingeleitet. Mehrere Optionen dürfen meist zusammengezogen werden, z. B. `ls -ali` statt `ls -a -l -i`. Die meisten Kommandos haben ein Dutzend Optionen, die Sie sich niemals alle merken können (wir übrigens auch nicht). Deshalb gibt es die eingebaute Hilfe in Form des Kommandos `man` (für „Manual“). `man ls` liefert beispielsweise alles Wissenswerte über das Kommando `ls`. Mehr dazu später.

Gelegentlich werden Sie bei Kommandozeilen weiter hinten ein zweites #-Zeichen finden. Dieses Zeichen kennzeichnet einen Kommentar. Wenn Sie Kommandos eingeben, brauchen Sie Kommentare nicht mit einzugeben (obwohl es auch nicht schadet). Die Doppelbedeutung von # als `root`-Prompt und als Kommentarzeichen kann manchmal Verwirrung stiften, insbesondere, wenn Listings abgedruckt sind, die mit Kommentaren eingeleitet werden. Die richtige Bedeutung des Zeichens geht aber aus dem Zusammenhang immer eindeutig hervor. Generell gilt, dass nur die fett hervorgehobenen Zeichen einzugeben sind!

Auf Ihrem Rechner wird vor dem # oder $ meist noch das aktuelle Verzeichnis und eventuell der Rechnername angegeben. Auf diese Angaben wird in diesem Buch aus Gründen der Übersichtlichkeit generell verzichtet. Die folgenden Zeilen (bis wieder eine Zeile mit `root#` oder `user$` beginnt) sind das Ergebnis des Kommandos.

Manchmal reicht der Platz in diesem Buch nicht aus, um ein Kommando in einer einzigen Zeile abzudrucken. In solchen Fällen wird das Kommando über mehrere Zeilen verteilt, die durch das Zeichen \getrennt sind. Das sieht dann beispielsweise so aus:

```
root#  find /home -group users \
$¿$        -ctime -5
```

Sie können dieses Kommando nun ebenfalls zweizeilig eingeben – dann müssen Sie die erste Zeile wie im Buch mit \und ↵ abschließen. Sie können die zwei Zeilen aber auch einfach zusammenziehen – dann entfällt das Zeichen \! Generell hebt das Zeichen \die Sonderbedeutung des nachfolgenden Zeichens auf (in obigem Fall wird „Eingabeende" zu „Neue Zeile").

Programme im Hintergrund ausführen: Sie können Programme auch im Hintergrund ausführen. Sie müssen nicht auf das Programmende warten, sondern können sofort weiterarbeiten. Dazu geben Sie am Ende der Kommandozeile das Zeichen & an. Diese Vorgehensweise empfiehlt sich vor allem, wenn Sie aus einer Konsole heraus ein Programm mit grafischer Benutzeroberfläche starten (z. B. `mozilla &`).

Wenn Sie das Zeichen & vergessen haben, ist das auch kein Malheur. Sie können das Programm auch nachträglich in den Hintergrund befördern. Dazu unterbrechen Sie das Programm mit Strg+Z und führen anschließend das Kommando `bg` aus.

Liste der aktiven Prozesse anzeigen: Als „Prozess" bezeichet man oft ein Programm, das sich im Arbeitsspeicher befindet und gerade läuft (im Gegensatz zu einem Programm, das als Datei auf der Platte ruht). Mit `ps` können Sie eine Liste Ihrer zurzeit aktiven Prozesse samt ihrer Prozessnummern anzeigen; `ps ax` liefert eine erweiterte Liste aller Prozesse. Komfortabler geht es mit `top`, das eine Liste der Prozesse liefert, die alle fünf Sekunden aktualisiert wird, bis Sie `top` durch Q beenden. Die Prozessliste ist so geordnet, dass jene Programme oben angezeigt werden, die momentan die meiste Rechenleistung benötigen.

Aktiven Prozess löschen: Mit dem Kommando `kill` <`Signalnummer`> <`Prozessnummer`> kann man einen Prozess durch Angabe seiner Prozessnummer löschen. Wird keine Signalnummer angegeben, wird der Wert für „Terminieren" verwendet. Der Prozess kann dann noch offene Dateien schliessen und auch sonst aufräumen. Manchmal zeigen sich Prozesse aber renitent. Dem Signal „kill" mit Nummer 9 kann sich jedoch kein Prozess verweigern (z. B.: `kill -9 1234`). Gibt man als Prozessnummer die 0 an, werden alle eigenen Prozesse gelöscht (probieren Sie doch mal (z. B.: `kill -9 0`, aber bitte nicht als `root` (was passiert, wenn doch?)).

Im Programm `top` genügt die Taste K (*kill*) und die Angabe der Prozessnummer, um einem Prozess zu beenden. Das ist natürlich nur in Ausnahmefällen sinnvoll. Aus Sicherheitsgründen können Sie nur solche Programme stoppen, die Sie selbst gestartet haben. (Nur `root` darf alle Programme beenden.)

Dämonen

Als „Dämon-Prozesse" (engl. daemons) werden Hintergrundprozesse zur Systemverwaltung bezeichnet. Diese Prozesse werden während des Hochfahrens des Rechners gestartet. Den Prozessen ist kein Terminal zugeordnet, weswegen in der TTY-Spalte des `ps`-Kommandos nur ein Fragezeichen angezeigt wird.

Im Folgenden wird ganz kurz die Bedeutung einiger besonders wichtiger Linux-Dämonen beschrieben:

`crond`	Batchdämon (startet andere Prozesse zu vorgegebenen Zeiten); die Steuerung erfolgt durch `/etc/crontab` sowie durch benutzerspezifische Dateien in `/var/cron/tabs`.
`inetd`	Startet je nach Bedarf andere Netzwerkdämonen; Steuerung durch `/etc/inetd.conf`.
`klogd`	Protokollierung von Kernel-Meldungen (üblicherweise in `/var/adm/*` oder `/var/log/*`)
`lpd`	Druckerspooler
`syslogd`	Protokollierung von Systemmeldungen (üblicherweise in `/var/adm/*` oder `/var/log/*`); Steuerung durch `/etc/syslog.conf`

Auch wenn auf die Steuerung der Dämonen in diesem Buch nicht detailliert eingegangen wird, sind auch für Anwender ohne allzu tief reichende Systemkenntnisse einige Zusatzinformationen ganz nützlich.

Mit dem `crond` lassen sich stets wiederkehrende Aktionen automatisieren. Wenn Ihr Rechner also plötzlich scheinbar unmotiviert mit der Festplatte rappelt, Ihnen E-Mails zusendet oder sonst etwas treibt, während Sie gerade Ihren dritten Kaffee zu sich nehmen, dann ist die Ursache fast immer der automatische Start von Prozessen durch `crond`. Bei manchen Distributionen ist hier mehr vorkonfiguriert, als für eine Nutzung von Linux auf einem Stand-Alone-Rechner sinnvoll ist. Werfen Sie einen Blick in die systemweite Datei `/etc/crontab`, in die Dateien in den Verzeichnissen `/etc/cron.*` sowie in die benutzerspezifischen Dateien in `/var/cron/tabs/*`.

Um die globale `cron`-Konfiguration zu verändern, können Sie `/etc/crontab` bzw. die Dateien in `/etc/cron*` direkt mit einem Editor bearbeiten. Wenn Sie dagegen benutzerspezifische `cron`-Einträge vornehmen möchten, sollten Sie dazu das Kommando `crontab -e` einsetzen. (Führen Sie vorher `export EDITOR=vi` aus, wenn Sie mit dem `vi` arbeiten möchten. Die Manual-Seiten zu `cron` und `crontab` geben weitere Informationen.)

Tipp

Bei manchen Distributionen wird ein grafisches Tool zur Bearbeitung der `cron`-Konfigurationsdateien mitgeliefert: `gcrontab` (Gnome), `kcrontab` und `KCron` (beide KDE) und `vcron` (Tcl/Tk). Die genannten Programme bieten mehr Komfort als `crontab`, sind allerdings noch nicht gleichermaßen ausgereift und stabil.

Die beiden Log-Dämonen `klogd` und `syslogd` protokollieren eine Menge wichtiger Meldungen des Kernels und diverser anderer Programme in Log-Dateien. Wenn es beim Betrieb von Linux Probleme gibt, sind die darin enthaltenen In-

formationen unverzichtbar und weisen oft auf die Ursache des Problems hin. die Logdateien finden Sie alle unterhalb des `/var`-Verzeichnisses.

Die Log-Dateien wachsen allerdings stetig. Daher ist es hin und wieder notwendig, sie zu verkürzen. Je nach Konfiguration kann das automatisch durch `crond` erfolgen (z. B. einmal pro Woche), automatisch bei jedem Systemstart oder aber manuell (einfach löschen).

Neben `crond` und den Log-Dämonen existieren unzählige weitere, die unter anderem Netzwerkdienste (ftp, http, NFS) zur Verfügung stellen, Kernel-Module dynamisch laden bzw. entfernen etc.

3.4.4 Arbeiten als root

Wie in diesem Kapitel bereits erwähnt wurde, sollten Sie **nur dann** als `root` arbeiten, wenn es unumgänglich ist (Durchführen administrativer Tätigkeiten). Auch wenn Sie als gewöhnlicher Benutzer eingeloggt sind, können Sie bequem und rasch ein Kommando als `root` ausführen: Führen Sie in der aktuellen Konsole bzw. im Shell-Fenster einfach `su -` aus. Damit loggen Sie sich als `root` ein. (Dazu müssen Sie natürlich das `root`-Passwort kennen.) Nun können Sie als `root` textorientierte Kommandos ausführen. `exit` oder Strg+D führt wieder zum ursprünglichen User zurück.

Alternativ kann das Kommando `sudo` verwendet werden. Hier lassen sich die Zugriffsrechte auf `root`-Kommandos begrenzen. Geregelt wird dies in der Datei `/etc/sudoers`, die mit dem Kommando `visudo` bearbeitet wird. Der grosse Vorteil bei `sudo` ist, dass man als Passwort das eigene Kennwort eingeben muss.

Falls Sie nach `su -` ein X-, KDE- oder Gnome-Programm ausführen möchten, wird es unter Umständen etwas komplizierter. Je nach Konfiguration treten nun nämlich die Fehler *unable to open display* oder *connection to localhost:0 refused by server* auf. Die Ursache für die Fehler ist ein Sicherheitsmechanismus, der fremden Anwendern das Ausführen von Programmen unter X verbietet. Wir kommen auf die Lösung dieses Problems im Kapitel über die grafische Oberfläche zurück.

3.5 Umgang mit Dateien und Verzeichnissen

Dieser Abschnitt beleuchtet unterschiedliche Aspekte der Dateiverwaltung unter Linux. Der Abschnitt soll Ihnen dabei helfen, das Linux-Dateisystem nicht nur zu verwenden, sondern vor allem auch verstehen zu lernen. Die wichtigsten Themen sind:

- Dateien und Verzeichnisse (Dateinamen, Jokerzeichen * und ?, Verzeichnisse)
- Linux-Verzeichnisstruktur (Wo befinden sich welche Dateien? Wofür stehen die Kürzel `bin`, `sbin`, `lib`, `usr`, `home` etc.?)

- Links (Feste und symbolische Verweise auf andere Dateien und Verzeichnisse)

3.5.1 Dateien

Ganz kurz die wichtigsten Fakten zu Dateinamen:

- Unter Linux sind Dateinamen mit einer Länge bis zu 255 Zeichen zulässig.
- Es wird zwischen Groß- und Kleinschreibung unterschieden! `readme`, `Readme` und `README` bezeichnen drei verschiedene Dateien!
- Internationale Zeichen im Dateinamen sind zwar im Prinzip zulässig, können aber zu Problemen führen und sollten daher nur mit Vorsicht verwendet werden. Aus der Sicht des Linux-Kernels ist der Dateiname einfach eine Bytefolge, in der lediglich das Zeichen / und der Code 0 nicht vorkommen dürfen.

  Das Problem ist aber, welcher Zeichensatz bei der Eingabe bzw. Anzeige von Dateinamen gilt und welcher Zeichensatz bei der Speicherung des Dateinamens verwendet wird. Hier gelten je nach Distribution unterschiedliche Einstellungen. So verwenden z. B. Red Hat und Fedora per Default den UTF8-Zeichensatz (Unicode), während bei Suse normalerweise der Latin-1-Zeichensatz oder dessen eurokompatible Variante Latin-0 aktiv ist. Wenn Sie mehrere Betriebssysteme parallel oder in einem gemeinsamen Netzwerk verwenden, ist es oft sinnvoll, sich auf den kleinsten gemeinsamen Nenner zu beschränken, d. h. nur US-ASCII-Zeichen in Dateinamen zu verwenden.
- Dateien, die mit einem Punkt beginnen, gelten als versteckte Dateien.

Der letzte Punkt bedarf noch einiger Anmerkungen: Sie können in Dateinamen beliebig viele Punkte verwenden. Der Punkt ist in Linux-Dateinamen ein Zeichen wie jedes andere, auch wenn er häufig zur Trennung von Namensteilen verwendet wird. **`utils-2.3.47.tar.gz`** ist ein ganz normaler Dateiname, der andeutet, dass es sich um eine komprimierte Datei mit irgendwelchen Utilities handelt. Nur wenn der Punkt als erstes Zeichen eines Dateinamens verwendet wird, hat er eine besondere Bedeutung: Die Datei gilt dann als **versteckt** und wird üblicherweise durch das Kommando **`ls`** nicht angezeigt.

Dateinamen, die bei der Eingabe von Kommandos nicht eindeutig als solche erkennbar sind (etwa Dateinamen mit Leerzeichen), müssen in Hochkommata gestellt werden (etwa **`"a b"`**).

3.5.2 Kommandos zur Bearbeitung von Dateien

Obwohl unter KDE und Gnome moderne Dateimanager zur Verfügung stehen, verwenden erfahrene Linux-Anwender oft noch immer diverse, textorientierte Kommandos, von denen hier die wichtigsten aufgezählt sind. (Für Leute, die sich noch mit DOS auskennen, sind in Klammern die äquivalenten DOS-Kommandos angegeben.)

`cd`	wechselt das aktuelle Verzeichnis (CHDIR)
`cp`	kopiert Dateien (COPY)
`less`	zeigt Textdateien seitenweise an (CAT, MORE)
`ls`	zeigt alle Dateien eines Verzeichnisses an (DIR)
`mkdir`	erzeugt ein neues Verzeichnis (MKDIR)
`mv`	verschiebt Dateien bzw. ändert ihren Namen (MOVE)
`rm`	löscht Dateien (DEL)
`rmdir`	löscht Verzeichnisse (RMDIR)

Das Inhaltsverzeichnis des aktuellen Verzeichnisses kann also mit `ls` angezeigt werden. Mehr Informationen erhalten Sie mit `ls -l`. Dateinamen, die mit einem Punkt beginnen, werden von `ls` normalerweise nicht angezeigt, sie sind also unsichtbar. Aus diesem Grund werden solche Dateinamen oft für Konfigurationsdateien verwendet. Damit `ls` auch diese Dateien anzeigt, muss zusätzlich die Option `-a` verwendet werden.

Mit `cp` können Sie Dateien kopieren, mit `mv` umbenennen und mit `rm` löschen. Dabei ist auch die Angabe von Joker-Zeichen erlaubt: `*` steht für mehrere beliebige Buchstaben, `?` für einen beliebigen Buchstaben (siehe unten).

Wenn Sie eine Datei suchen, verwenden Sie am einfachsten das Kommando `find -name '*name*'`. Linux durchsucht dann den gesamten Verzeichnisbaum ab dem gerade aktuellen Verzeichnis nach einem Dateinamen, in dem `name` vorkommt. Wenn Sie vorher in das Wurzelverzeichnis `/` wechseln, wird das gesamte Dateisystem durchsucht. Das kann ziemlich lange dauern, besonders wenn ein CD-ROM-Laufwerk eingebunden ist. Speziell beim Suchen von Dateien helfen die folgenden Kommandos:

`find`	sucht Dateien nach Name, Datum, Größe etc.
`locate`	sucht Dateien
`updatedb`	aktualisiert die Suchdatenbank für `locate`
`whereis`	sucht Programme in typischen `bin`-Verzeichnissen
`which`	sucht Programme in `PATH`-Verzeichnissen

Wenn Sie wissen möchten, wie viel Platz die Dateien innerhalb eines Verzeichnisses beanspruchen oder wie viel Platz auf Ihrer Festplatte noch frei ist, sollten Sie sich mit diesen Kommandos anfreunden:

`df`	zeigt den freien Speicher auf der Festplatte an
`du`	ermittelt den Platzbedarf eines Verzeichnisses

Ein Sonderfall ist der Umgang mit Windows-Disketten (also Disketten, die mit einem Windows-Dateisystem formatiert sind). Sie können solche Disketten wie Festplattenpartitionen in den Linux-Verzeichnisbaum einbinden und auf die Dateien dann ganz normal zugreifen. Wenn Sie rasch nur eine Datei von der oder auf die Diskette kopieren möchten, sind die so genannten `mtools` bequemer. Sie ermöglichen einen direkten Zugang zu Disketten:

`mcopy`	kopiert Dateien von bzw. nach Linux
`mdel`	löscht Dateien
`mdir`	zeigt das Inhaltsverzeichnis an
`mmd`	erzeugt ein neues Verzeichnis
`mrd`	löscht Verzeichnisse
`mren`	ändert den Namen von Dateien

Tipp

Eine Beschreibung dieser und vieler weiterer Kommandos zum Umgang mit Dateien und Verzeichnissen würde den Rahmen dieses Buchs sprengen. Wenn Sie rasch Informationen zu den Optionen eines Kommandos benötigen, geben Sie am einfachsten `man kommando` ein, also beispielsweise `man ls`.

3.5.3 Textdateien anzeigen

Unter KDE oder Gnome können Sie Textdateien direkt im Dateimanager (Konqueror oder Nautilus) lesen. Mit der rechten Maustaste können Sie die Datei auch in einem komfortablen Editor öffnen.

Arbeiten Sie in einer Textkonsole oder in einem Shell-Fenster, verwenden Sie zum Betrachten von Dateien am besten das Kommando `less`. Das Kommando kann auch mit anderen Kommandos verknüpft werden, um deren oft seitenweise Ausgaben in Ruhe zu lesen:

```
user$ less datei     # seitenweise Anzeige der Datei
user$ ls -l — less    # seitenweise Anzeige des Dateiverzeichnisses
```

Solange `less` läuft, können Sie sich mit den Cursortasten durch den Text bewegen. Die Leertaste erlaubt seitenweises Blättern, mit Backspace rückwärts. Mit (/) können Sie nach einem bestimmten Text suchen. (N) wiederholt die Suche. (Q) beendet `less`.

> **Tipp**
> Wenn Sie in einer Textkonsole mit `less` eine Datei anzeigen, die statt Text binäre Daten enthält, kann es passieren, dass die Daten als Sonderzeichen interpretiert werden und die Konsole dabei durcheinander kommt. In diesem Fall werden nur noch seltsame Zeichen am Bildschirm angezeigt, d. h. die Zuordnung des Zeichensatzes stimmt nicht mehr. Um Abhilfe zu schaffen, führen Sie das Kommando `reset` aus.

Ein älterer Verwandter von `less` ist `more`, dessen Bedienung ähnlich funktioniert – nur gibt es weniger Befehle im interaktiven Modus.

3.5.4 Texteditoren

Solange Sie mit KDE oder Gnome arbeiten, stehen Ihnen zahlreiche komfortable Programme zum Anzeigen bzw. Verändern von Textdateien zur Verfügung – unter KDE beispielsweise `kate` und `kwrite`, unter Gnome etwa `gedit`. Die Bedienung dieser Programme ist so intuitiv, dass hier keine weitere Beschreibung erforderlich ist.

Darüber hinaus gibt es aber auch zahlreiche Editoren, die in einer Textkonsole bzw. in einem Shell-Fenster ausgeführt werden können. Das ist besonders dann praktisch, wenn X aus irgendeinem Grund nicht verwendet werden kann. Sie heißen unter anderem „Emacs“, „Jove“, „Jed“, „Pico“ oder „vi“. Jeder Linux-User hat hier seine Vorlieben. Wir behandeln hier nur den „Großvater“ aller Editoren, der immer und überall verfügbar ist.

vi, vim und elvis

`vim` und `elvis` sind zwei `vi`-kompatible Editoren. Der Original-`vi` ist aus urheberrechtlichen Gründen nicht Teil von Linux. Das Kommando `vi` kann aber zumeist dennoch ausgeführt werden und bewirkt dann automatisch den Start von `vim` oder `elvis`.

Die Bedienung des `vi` ist (höflich formuliert) etwas unkonventionell. (`vi`-Profis behaupten freilich hartnäckig, dass es keinen besseren Editor gebe.) Obwohl der Editor für Linux-Einsteiger also kaum die erste Wahl darstellt, ist eine kurze Beschreibung hier dennoch angebracht:

- Der `vi` ist vergleichsweise kompakt und steht eventuell auch dann zur Verfügung, wenn kein anderer Editor läuft (etwa bei Wartungsarbeiten, wenn Linux von einer Notfalldiskette gebootet wurde).
- Der `vi` steht unter praktisch allen Unix-Systemen zur Verfügung und stellt insofern einen Standard dar. Aus diesem Grund wird der `vi` von diversen Programmen automatisch als Editor aufgerufen, wenn dies nicht explizit durch eine entsprechende Konfiguration verhindert wird. In solchen Fällen müssen Sie zumindest wissen, wie Sie den Editor wieder verlassen können.

Der fundamentale Unterschied zu anderen Editoren besteht also darin, dass der `vi` zwischen verschiedenen Modi unterscheidet. Die Texteingabe ist nur im Insert-Modus möglich. Die Eingabe der meisten Kommandos erfolgt im Complex-Command-Modus, der mit (:) aktiviert wird. Vorher muss gegebenenfalls der Insert-Modus durch (Esc) verlassen werden. Abschließend folgt hier noch eine Referenz der elementaren Kommandos:

vi-Tastenkürzel

(I)	wechselt in den Insert-Modus
(Esc)	beendet den Insert-Modus
(H) / (L)	Cursor-Bewegung links/rechts
(J) / (K)	Cursor-Bewegung ab/auf
(X)	löscht ein Zeichen
(D) (D)	löscht die aktuelle Zeile
(P)	gelöschte Zeile an der Cursor-Position einfügen
(U)	generelles Undo
(:)	wechselt in den Complex-Command-Modus

Kommandos im Complex-Command-Modus

:w *name*	speichert den Text unter einem neuen Namen
:wq	speichert und beendet den `vi`
:q!	beendet den `vi`, ohne zu speichern
:help	startet die Online-Hilfe

3.5.5 Jokerzeichen

Im täglichen Umgang mit Dateien werden Sie häufig ganze Gruppen von Dateien bearbeiten – etwa alle Dateien mit der Endung `.tex`. Um das zu ermöglichen, sind bei der Eingabe von Linux-Kommandos zwei Jokerzeichen erlaubt: `?` zur Spezifikation *eines* beliebigen Zeichens und `*` zur Spezifikation beliebig vieler (auch null) Zeichen.

Auf den ersten Blick sieht das genauso aus wie unter DOS/Windows. Der wohl wichtigste Unterschied besteht darin, dass `*` fast alle Zeichen erfasst, also auch Punkte (sofern sie nicht am Beginn des Dateinamens stehen). Wenn Sie alle Dateien erfassen möchten, heißt es unter Linux `*` und nicht `*.*`! (Anmerkungen zu versteckten Dateien folgen weiter unten.)

Ein weiterer Unterschied besteht darin, dass auch mehrere Jokerzeichen Linux nicht aus dem Gleichgewicht bringen. Sie können beispielsweise mit `*graf*` alle Dateien suchen, die `graf` in ihrem Namen enthalten – also etwa `grafik.doc`, `apfelgraf` und `README.graf`.

Wenn Ihnen die Jokerzeichen `?` und `*` zu allgemein sind, können Sie eine stärkere Einschränkung durch die Angabe eckiger Klammern erreichen. `[abc]` steht

als Platzhalter für einen der drei Buchstaben `a`, `b` oder `c`. Wenn innerhalb der eckigen Klammern ein Bindestrich zwischen zwei Buchstaben oder Ziffern angegeben wird, dann ist ein Zeichen dazwischen gemeint: `[a-f]*` erfasst demnach alle Dateien, die mit einem Buchstaben zwischen `a` und `f` beginnen. `*[_.-]*` meint alle Dateien, die irgendwo in ihrem Dateinamen zumindest einen Punkt, Unterstrich oder Bindestrich enthalten. Durch ein Ausrufezeichen kann der Ausdruck negiert werden: `[!a-z]*` meint alle Dateien, die mit einem Großbuchstaben oder mit einem Sonderzeichen beginnen. `*.[hc]` erfasst alle Dateien, die mit `.c` oder `.h` enden.

Die Jokerzeichen können auch für Verzeichnisse verwendet werden. `*/*.tex` erfasst alle `*.tex`-Dateien, die sich in Unterverzeichnissen des aktuellen Verzeichnisses befinden (nur eine Ebene, also nicht auch Dateien in Unter-Unterverzeichnissen). `/usr/*bin/*` erfasst alle Dateien in den Verzeichnissen `/usr/bin` und `/usr/sbin`.

Beim Kopieren von Dateien wird oft ein weiteres Sonderzeichen benötigt – der Punkt als Platzhalter für das aktuelle Verzeichnis. So werden durch das Kommando `cp projekt/*.c .` alle `*.c`-Dateien aus dem Verzeichnis `project` in das aktuelle Verzeichnis kopiert.

Jokerzeichen für Dateinamen	
`?`	genau ein beliebiges Zeichen
`*`	beliebig viele (auch null) beliebige Zeichen
`[abc]`	genau eines der angegebenen Zeichen
`[a-f]`	ein Zeichen aus dem angegebenen Bereich
`[!abc]`	keines der angegebenen Zeichen
`[^abc]`	wie oben
`~`	Abkürzung für das Heimatverzeichnis

Im Umgang mit Verzeichnissen sind außerdem die beiden Dateinamen `.` und `..` von besonderer Bedeutung. `.` verweist auf das aktuelle Verzeichnis, `..` auf das übergeordnete Verzeichnis.

Hinweis

Für die Auswertung der Jokerzeichen ist nicht das jeweils aufgerufene Kommando zuständig, sondern die Shell, aus der das Kommando aufgerufen wird. `bash`, die unter Linux gebräuchlichste Shell, kennt neben den gerade beschriebenen Jokerzeichen eine Menge weiterer Sonderzeichen, die bei der Ausführung eines Kommandos eine besondere Wirkung haben (siehe Kapitel 4).

3.5.6 Komplikationen bei der Verwendung von Jokerzeichen

Der Umgang mit Jokerzeichen sieht auf den ersten Blick einfacher aus, als er in Wirklichkeit ist. Wenn Sie Schwierigkeiten mit Jokerzeichen haben, sollten Sie einfach einige Experimente mit `echo jokerzeichen` durchführen. Dieses Kommando zeigt einfach alle durch eine Jokerzeichen-Kombination erfassten Dateinamen auf dem Bildschirm an, ohne die Dateinamen zu verändern.

Ein Problem besteht darin, dass `*` nicht nur Dateien, sondern auch Verzeichnisse erfasst. `ls *` zeigt aus diesem Grund nicht nur alle Dateien im aktuellen Verzeichnis an, sondern auch den Inhalt aller Unterverzeichnisse, die über `*` erfasst werden. Beim Kommando `ls` kann dieses Problem durch die Option `-d` umgangen werden; bei anderen Kommandos steht diese Option allerdings nicht zur Verfügung.

Wenn Sie alle Verzeichnisse (nicht aber normale Dateien) bearbeiten möchten, hilft die Jokerzeichenkombination `*/.` weiter: Mit ihr werden alle „Dateien" erfasst, die als Unterverzeichnis einen Verweis auf sich selbst enthalten – und das ist eben nur bei Verzeichnissen der Fall. (Verzeichnisse gelten intern als eine Sonderform einer Datei – daher die Anführungszeichen.)

```
user$  echo */.
```

Die Tatsache, dass nicht das jeweilige Programm, sondern schon die Shell für die Verarbeitung der Jokerzeichen zuständig ist, hat nicht nur Vorteile. So ist es etwa unmöglich, mit `ls -R *.tex` nach allen `*.tex`-Dateien auch in Unterverzeichnissen zu suchen. (`-R` = rekursives Durchsuchen von Unterverzeichnissen.)

Der Grund dafür ist recht einfach: Die Shell erweitert das Muster `*.tex` für das *aktuelle* Verzeichnis und übergibt die Liste der gefundenen Dateien an `ls`. Das Kommando zeigt Informationen zu diesen Dateien an. Wenn Sie keine Verzeichnisse mit der Endung `.tex` haben, ist `ls` damit am Ende – auch die Option `-R` kann daran nichts mehr ändern. Rekursiv durchsucht werden nämlich nur die Verzeichnisse, die als Parameter übergeben werden.

Zum Suchen nach Dateien stellt Linux deshalb das sehr viel flexiblere Kommando `find` zur Verfügung. Im Beispiel unten wird eine Liste aller `*.tex`-Dateien im aktuellen und in allen untergeordneten Verzeichnissen angezeigt:

```
user$  find . -name '*.tex'
```

Dieselbe Ursache hat auch das Scheitern des Versuchs, `*~`-Dateien in allen Unterverzeichnissen zu löschen. Das Kommando `rm -r *~` versagt trotz der Option `-r` für ein rekursives Bearbeiten von Unterverzeichnissen. Der Grund: `*~` wird bereits von der `bash` erweitert und zwar für das aktuelle Verzeichnis. Unterverzeichnisse werden nur berücksichtigt, wenn sie einen Namen aufweisen, der mit `~` endet. (Beabsichtigt ist ja genau das Gegenteil, nämlich Dateien mit `~` in gewöhnlichen Verzeichnissen zu löschen.)

Das Löschen aller Backup-Dateien in Unterverzeichnissen gelingt erst mit dem folgenden Kommando. Dabei wird die Liste aller in Frage kommenden Dateien mit `find` gebildet und via `xargs`-Kommando an `rm` weitergeleitet.

```
user$ find . -name '*~' | xargs rm
```



Eines der gefährlichsten Linux-Kommandos ist `rm -rf *`: Damit werden rekursiv alle Dateien und alle Verzeichnisse (ausgehend vom aktuellen Verzeichnis) gelöscht.

Leider ist es in Linux nicht möglich, mit dem Kommando `mv *.x *.y` alle `*.x`-Dateien in `*.y`-Dateien umzubenennen. Der Grund für diese Einschränkung ist wieder derselbe, wie oben beschrieben: Die Shell ersetzt `*.x` durch die Liste aller Dateien, die diesem Muster entsprechen. Für `*.y` gibt es keine gültigen Dateinamen. An das Kommando `mv` wird daher eine Liste mehrerer Dateien und der Ausdruck `*.y` übergeben – und `mv` weiß dann nicht, was es mit diesen Argumenten tun soll. Das Kommando ist prinzipiell nicht in der Lage, mehrere Dateien auf einmal umzubenennen. Entweder werden *mehrere* Dateien in ein anderes Verzeichnis verschoben oder es wird nur *eine* Datei umbenannt.

3.5.7 Versteckte Dateien

Unter Linux gelten Dateien, deren Name mit einem Punkt beginnt, als versteckte Dateien. `*` berücksichtigt deswegen nicht wirklich alle Dateien in einem Verzeichnis: Dateien, die mit einem Punkt beginnen (häufig Konfigurationsdateien, die man gar nicht sehen will), werden ignoriert.

Wenn Sie nun glauben, Sie könnten unsichtbare Dateien mit `.*` erfassen, wird alles noch schlimmer: Damit sind nämlich nicht nur unsichtbare Dateien gemeint, die mit `.` beginnen, sondern auch die Verzeichnisse `.` und `..` (also das aktuelle und das übergeordnete Verzeichnis). Wenn das jeweilige Kommando in der Lage ist, ganze Verzeichnisse zu bearbeiten, können die Folgen fatal sein.

Das Problem kann mit dem Suchmuster `.[!.]*` umgangen werden. Damit werden alle Dateinamen erfasst, deren erstes Zeichen ein Punkt ist, die mindestens ein weiteres Zeichen aufweisen, das kein Punkt ist, und die beliebig viele (auch null) weitere Zeichen haben.

```
user$ echo .[!.]*
```

Beim Kommando `ls` kann die Option `-a` verwendet werden. Sie führt dazu, dass alle Dateien (auch unsichtbare) angezeigt werden. Allerdings dürfen bei dieser Verwendung von `ls` keine Masken angegeben werden. `-a` funktioniert nur dann, wenn `ls` sich die Dateien selbst suchen darf und nicht die Shell diese Aufgabe übernimmt.

Wirklich universell funktioniert auch in diesem Fall nur `find`. Das folgende Kommando findet alle versteckten Dateien im aktuellen Verzeichnis und allen untergeordneten Verzeichnissen:

```
user$ find . -name '.*'
```

3.5.8 Verzeichnisse

Mit `cd` und darauffolgender Angabe eines Verzeichnisses können Sie das aktuelle Verzeichnis wechseln. `pwd` zeigt das aktuelle Verzeichnis an. Ein wesentlicher Unterschied zwischen Windows und Linux besteht darin, dass als Trennzeichen zwischen den Verzeichnissen `/` (und nicht \) verwendet wird! Auch hat bei Windows das Kommando `cd` ohne Parameter die Funktion des Linux-`pwd`. `cd` ohne Parameter führt Sie bei Linux in Ihr Heimatverzeichnis.

Heimatverzeichnis: Nach dem Einloggen befinden Sie sich automatisch in einem Verzeichnis, das Ihnen gehört. Dieses Verzeichnis wird Heimat- oder Home-Verzeichnis genannt. Alle darin enthaltenen Dateien und Unterverzeichnisse gehören Ihnen. Andere Benutzer (mit Ausnahme von `root`) dürfen diese Dateien weder verändern noch löschen (und je nach Einstellung nicht einmal lesen).

Das Heimatverzeichnis wird mit der Tilde `~` abgekürzt. Bei `root` heißt das Heimatverzeichnis `/root`. Bei fast allen anderen Linux-Anwendern befindet sich das Heimatverzeichnis in `/home/name`. Mit `cd` ohne Angabe weiterer Parameter wechseln Sie in das Heimatverzeichnis zurück.

Verzeichnisbaum: Das Dateisystem beginnt mit dem Wurzelverzeichnis `/`. Auch wenn es mehrere Festplatten bzw. Festplattenpartitionen sowie CD- und DVD-Laufwerke gibt, sind alle Daten in den Verzeichnisbaum eingebunden. Laufwerksangaben wie `C:` sind unter Linux weder möglich noch sinnvoll. Innerhalb dieses Buchs gelten alle weiteren Verzeichnisse als *untergeordnet*, das Wurzelverzeichnis steht also – bildlich gesehen – ganz oben. Beispielsweise kann der Inhalt eines CD-Laufwerks üblicherweise unter dem Verzeichnis `/mnt/cdrom` oder `/media/cdrom` gelesen werden. Aus diesem Grund besteht unter Linux keine Notwendigkeit für die Laufwerksbuchstaben `A:`, `C:` etc., die unter Windows üblich sind. Auch alle Geräte werden auf das Dateisystem abgebildet und können wie Dateien angesprochen werden.

Der Punkt `.` ist bei Linux (wie bei Windows/DOS) ein Verweis auf das aktuelle Verzeichnis. Er wird hier aber öfter gebraucht, denn es muss beim Kopieren immer ein Zielverzeichnis angegeben werden. Bei Windows/DOS konnte man diese Angabe weglassen, wenn das aktuelle Verzeichnis das Zielverzeichnis war.

Zwei Punkte `..` sind ein Verweis auf das übergeordnete Verzeichnis. Mit dem Kommando `cd ..` wandern Sie also im Dateibaum „nach oben“.

Eines der größten Probleme beim Neueinstieg in Unix/Linux besteht darin, eine bestimmte Datei im weit verästelten Verzeichnissystem zu finden. Eine erste

Orientierungshilfe bietet der folgende Abschnitt. Sehr zweckmäßig sind auch die Kommandos `find`, `locate` und `grep`: `find` sucht nach einem bestimmten Dateinamen und `grep` nach einer Zeichenkette innerhalb einer Datei. Durch die Kombination der beiden Kommandos können Sie mit etwas Übung beinahe jede Datei finden. `locate` funktioniert ähnlich wie `find`, setzt aber auf einer vorher erstellten Datenbank auf und ist daher ungleich schneller.

3.5.9 Feste und symbolische Links

Die Betriebssysteme verwalten Ihre Verzeichnisse und Dateien auf unterschiedliche Weise. Später werden Sie sehen, dass Linux sogar mehrere Möglichkeiten der Plattenverwaltung bietet. Beim Standard-Dateisystem besteht ein Verzeichniseintrag nur aus dem Namen der Datei und einem Verweis auf einen Verwaltungsblock („I-Node“ genannt), in dem alle Informationen zu einer Datei gespeichert werden (z. B. Eigentümer, belegte Plattenblöcke, Erstellungsdatum usw.). Diese I-Nodes werden einfach durchnummeriert. So ist es recht einfach, eine Datei zu verschieben – es muss nur der Namenseintrag von einem Verzeichnis ins andere kopiert werden, die Datei bleibt, wo sie ist. Diese Form der Speicherung ermöglicht auch die sogenannten Links.

Links sind Verweise auf Dateien. Durch Links kann von verschiedenen Orten in der Verzeichnisstruktur auf ein und dieselbe Datei zugegriffen werden, ohne dass diese Datei physikalisch mehrfach gespeichert werden muss. Links sind damit ein wichtiges Hilfsmittel zur Vermeidung von Redundanzen. Im Linux-Dateisystem kommen Links besonders häufig in `/bin`- und `/lib`-Verzeichnissen vor. (Sehen Sie sich beispielsweise `/usr/bin` oder `/usr/lib` mit `ls -l` genauer an!)

Am einfachsten sind Links anhand eines Beispiels zu verstehen: Angenommen, im Verzeichnis `test` befindet sich die Datei `abc`; durch das Kommando `ln abc xyz` wird scheinbar eine neue Datei `xyz` erstellt. In Wahrheit sind aber `abc` und `xyz` nur zwei Verweise auf ein und dieselbe Datei. Die einzige Möglichkeit, dies zu überprüfen, bietet das Kommando `ls -l`. Es gibt in der zweiten Spalte an, wie viele Links auf eine bestimmte Datei zeigen (im vorliegenden Beispiel zwei). Wenn zusätzlich die Option `-i` verwendet wird, gibt `ls` auch den I-Node der Datei an, der bei Links (und nur dann) identisch ist.

```
root# ls -li
 59293 -rw-r--r--   1 root     root    1004 Oct  4 16:40 abc
root# ln abc xyz
root# ls -li
 59293 -rw-r--r--   2 root     root    1004 Oct  4 16:40 abc
 59293 -rw-r--r--   2 root     root    1004 Oct  4 16:40 xyz
```

Über beide Namen greifen Sie auf ein und dieselbe Datei zu, etwa beim Ändern. Wenn Sie einen der beiden Dateinamen löschen, reduzieren Sie dadurch nur die Anzahl der Links. Ist diese Zahl auf 0 geschrumpft, putzt der Linux-Kernel auch die Datei selbst weg.

Symbolische Links: Linux kennt zwei Formen von Links. Das obige Beispiel hat feste Links (hard links) vorgestellt, wie sie standardmäßig durch das Kommando `ln` erzeugt werden. Wird `ln` dagegen mit der Option `-s` verwendet, erzeugt das Kommando symbolische Links. Symbolische Links (manchmal auch weiche Links bzw. soft links genannt) haben den Vorteil, dass sie innerhalb des Dateisystems von einer physikalischen Festplatte auf eine andere verweisen können und dass sie nicht nur auf Dateien, sondern auch auf Verzeichnisse angewandt werden können.

Durch `ls` wird bei symbolischen Links angezeigt, wo sich die Ursprungsdatei befindet. Es wird allerdings kein Zähler verwaltet, der angibt, von wie vielen Stellen auf die Ursprungsdatei verwiesen wird (er entspricht in etwa der „Verknüpfung" bei Windows).

Intern besteht der Unterschied zwischen festen und symbolischen Links darin, dass im einen Fall der I-Node, im anderen Fall der Dateiname oder (bei Links über ein Verzeichnis hinaus) die Pfadangabe gespeichert wird.

```
root#  ln -s abc efg
root#  ls -li
 59293 -rw-r--r--   2 root     root    1004 Oct  4 16:40 abc
 59310 lrwxrwxrwx   1 root     root       3 Oct  4 16:52 efg -> abc
 59293 -rw-r--r--   2 root     root    1004 Oct  4 16:40 xyz
```

Symbolische Links verhalten sich ein wenig anders als feste Links. Das Löschen der Ursprungsdatei (also `abc`) verändert den Link auf diese Datei nicht, `efg` verweist jetzt aber auf eine leere (gar nicht vorhandene) Datei. Wird dagegen der symbolische Link gelöscht, hat das keinen Einfluss auf die Ursprungsdatei.

Symbolische Links können nicht nur für Dateien, sondern auch für Verzeichnisse erstellt werden. Das kann einige Verwirrung stiften, weil durch einen symbolischen Link ganze Verzeichnisbäume scheinbar verdoppelt werden. In Wirklichkeit stellt der Verzeichnis-Link aber nur einen zusätzlichen Pfad zu denselben Dateien und Unterverzeichnissen dar.

Generell sollten Sie versuchen, möglichst keine absoluten, sondern nur relative Pfadangaben in Links zu verwenden. Damit vermeiden Sie Probleme, die sich beim Mounten von Verzeichnissen per NFS oder beim Verschieben von Verzeichnissen ergeben können.

Sowohl symbolische als auch feste Links haben Vorteile. Symbolische Links sind einfacher in der Handhabung. Dafür verbrauchen feste Links weniger Speicher und sind schneller.

3.5.10 Links auf Programme

Unter Linux kommt es relativ häufig vor, dass mehrere, oft ganz unterschiedlich arbeitende Kommandos durch symbolische oder feste Links auf ein und dasselbe Programm verweisen. Obwohl offensichtlich jedes Mal diese eine Programmdatei

ausgeführt wird, verhält sich das Programm ganz unterschiedlich, je nachdem, über welchen Link es aufgerufen wurde.

Die Erklärung ist recht einfach: Bei der Programmerstellung wurden diese unterschiedlichen Aufrufmöglichkeiten bereits eingeplant. Bei der Ausführung des Programms wird dann jedes Mal getestet, durch welchen Kommandonamen das Programm gestartet wurde. Wenn der Name erkannt wird, startet das Programm die entsprechenden Routinen, andernfalls liefert es eine Fehlermeldung.

Ein gutes Beispiel für diese Vorgehensweise bieten die `mtools`-Kommandos zur Bearbeitung von MS-DOS-Disketten. Die Kommandos `mattrib`, `mcd` etc. verweisen auf eine einzige Programmdatei, nämlich `mtools`. Wenn Sie versuchen, `mtools` direkt aufzurufen, liefert das Programm eine Fehlermeldung, weil der Kommandoname `mtools` in der Liste der dreizehn möglichen Kommandos nicht vorkommt.

3.6 Zugriffsrechte, Benutzer und Gruppenzugehörigkeit

Linux ist als Multiuser-System konzipiert und benötigt daher Mechanismen, die steuern, wer auf welche Dateien zugreifen darf, wer sie ändern darf etc. Die Basis des Zugriffssystems stellt die Verwaltung von Benutzern und Gruppen dar.

- Jeder Linux-Benutzer hat eine Benutzerkennung (user id, kurz: uid), mit der er sich gegenüber dem BS identifizieren kann.
- Jeder Linux-Benutzer gehört einer Gruppe an und besitzt damit eine Gruppen-ID, (kurz: gid).
- Jede Datei hat einen Eigentümer und eine Gruppe, die bei der Erzeugung der Datei eingetragen werden.
- Jede Datei besitzt zwölf voneinander unabhängige Schutzbits:

Special			User			Group			Others		
SUID	SGID	STI	R	W	X	R	W	X	R	W	X

Mit jeder Datei bzw. mit jedem Verzeichnis werden diese Informationen gespeichert.

Der Besitzer (*owner*) einer Datei ist in der Regel die Person, die die Datei erzeugt hat. Als Gruppe wird normalerweise die primäre Gruppe des Besitzers verwendet.

Die Zugriffsinformationen r, w und x steuern, wer die Datei lesen, schreiben (verändern) und ausführen darf. Diese Informationen werden getrennt für

den Besitzer, für die Gruppe und für alle anderen Benutzer gespeichert. Das ermöglicht es, dem Besitzer mehr Rechte zu geben als anderen Benutzern. Die Informationen werden meist Zugriffsbits genannt, weil sie intern als Zahl mit bitweiser Kodierung gespeichert werden. Die zwölf Zugriffsbits werden oft auch oktal dargestellt (siehe das Kommando `chmod`). Wir wollen zuerst die neun „normalen" Zugriffsrechte behandeln; auf die drei Spezialbits gehen wir später ein.

Die Zugriffsrechte, der Besitzer sowie die Gruppenzugehörigkeit einer Datei können mit `ls -l` betrachtet werden. Für eine typische Textdatei liefert `ls` das folgende Ergebnis: Die Datei darf vom Besitzer `michael` gelesen und verändert werden. Da es sich nicht um eine ausführbare Datei handelt, sind die Execute-Rechte deaktiviert. Alle anderen Benutzer, egal, ob sie Mitglied der `users`-Gruppe sind oder nicht, dürfen diese Datei lesen (aber nicht verändern). Für die drei Abteilungen Owner, Group und Others werden jeweils drei Buchstaben („r", „w" oder „x" für die Rechte angegeben, falls dieses Recht gesetzt ist, andernfalls wird an der entsprechenden Stelle ein „-" eingesetzt.

```
michael$  ls -l header.tex
-rw-r--r--   1 michael  users        3529 Oct  4 15:43 header.tex
```

Hinweis

`ls -l` zeigt nicht nur den Zustand der neun Zugriffsbits an, sondern sieht in der ersten Textspalte Platz für ein weiteres Zeichen vor. Darin wird der Dateityp angegeben (- für eine normale Datei, `d` für ein Verzeichnis (directory), `l` für einen symbolischen Link, `c` für ein zeichenorientiertes Gerät, `b` für ein blockorientiertes Gerät etc.).

Wenn `michael` möchte, dass diese Datei nur von den Mitgliedern der `users`-Gruppe, nicht aber von Anwendern außerhalb der Gruppe gelesen werden kann, dann muss er das letzte r-Bit deaktivieren. Dazu verwendet er das Kommando `chmod`.

```
michael$  chmod o-r header.tex
michael$  ls header.tex -l
-rw-r-----   1 michael  users        3529 Oct  4 15:43 header.tex
```

Beim Kommando `chmod` zum Einstellen der Zugriffsrechte einer Datei existieren zwei Modi, symbolisch oder direkt. Beim Eingeben des Kommandos kommt erst der Modus und dann die Liste der Dateien, deren Berechtigung geändert werden soll.

Symbolischer Modus: *Bereich Operand Berechtigung*
Der Bereich wird durch die Kombination der Buchstaben `u g o` festgelegt (`u` = User, `g` = Group, `o` = Others). Danach folgt der Operator. `+` bedeutet Hinzufügen des Zugriffsrechts, `-` das Entfernen und `=` setzt exakt die rechts daneben stehende Berechtigung. Die Berechtigungswerte sind `w` für Schreiben, `r` für Lesen und `x` für Ausführen. Daneben gibt es noch `s` für die unten näher erläuterten Spezialwerte SUID/SGID (je nachdem, ob `u+s` oder `g+s` angegeben wird) und `t` für das Sticky Bit (ohne Angabe von ugo). Beispiele:

```
chmod +x dat2            # Ausführungsrecht für alle hinzufügen
chmod go-w dat2          # Schreibrecht nur noch für User
chmod g+rwx dat3         # Volle Rechte für Gruppe
chmod ugo-rwx dat4       # Allen alle Rechte entziehen
chmod +t verz1           # Sticky-Bit von verz1 setzen
chmod u+s admintool      # SUID-Bit setzen
```

Direktmodus: *Oktalzahl (3–4 Stellen)*
Für *user, group* und *others* wird jeweils eine Oktalzahl angegeben, welche die jeweiligen Rechte wiedergibt. Dabei gilt folgende Wertigkeit:

`read`	4
`write`	2
`execute`	1

Falls die Zugriffsrechte SUID (4), SGID (2) und STICKY (1) vergeben werden sollen, kommt vorne noch eine vierte Stelle hinzu. Die Anwendung dieses Modus zeigen die folgenden Beispiele:

```
                     User   Group  Others
chmod 754 dat1       rwx    r-x    r--
chmod 740 dat2       rwx    r--    ---
chmod 440 wichtig    r--    r--    ---
chmod 000 geheim     ---    ---    ---
chmod 4711 systool   rws    --x    --x
```

Möglicherweise soll der Zugriff auf die Datei `header.tex` auf zwei Anwender beschränkt werden, `michael` und `kathrin`. Dazu kann eine neue Gruppe gebildet werden, der nur die beiden angehören. (Wenn `michael` und `kathrin` das Dokumentations-Team einer Firma bilden, wäre als Gruppenname etwa `dokuteam` sinnvoll. Anschließend wird die Gruppenzugehörigkeit mit `chgrp` geändert:

```
michael$  chgrp dokuteam header.tex
michael$  ls header.tex -l
-rw-r-----   1 michael  dokuteam     3529 Oct  4 15:43 header.tex
```

Hinweis

Die Kommandos `chgrp` (change group) zum Ändern der Gruppe einer Datei und `chown` (change owner) zum Ändern des Eigentümers dürfen in der Regel nur von `root` ausgeführt werden – alles andere wäre viel zu gefährlich!

3.6.1 Zugriffsrechte auf Verzeichnisse

Die neun Zugriffsbits haben im Prinzip auch bei Verzeichnissen Gültigkeit, allerdings besitzen sie dort eine etwas abweichende Bedeutung:

- Das r-Recht erlaubt es anderen Anwendern, den Inhalt des Verzeichnisses mit `ls` anzusehen,
- das x-Recht ermöglicht es darüber hinaus, mit `cd` in dieses Verzeichnis zu wechseln und
- das w-Recht erlaubt das Verändern des Verzeichnisinhalts – also das Löschen oder Anlegen von Dateien. Das gilt auch, wenn für die Datei kein Schreibrecht existiert.

Die Verzeichnis-Zugriffsrechte sind also wesentlich wichtiger (und gefährlicher) als die Datei-Zugriffsrechte. Deshalb ist beim Setzen von Zugriffsrechten immer die höchste Sorgfalt nötig.

3.6.2 Spezialbits

Die Bedeutung der drei mal drei Zugriffsbits rwxrwxrwx ist leicht zu verstehen. Darüber hinaus können mit Dateien und Verzeichnissen noch einige weitere Informationen gespeichert werden. Die Kenntnis dieser Spezialbits ist im Regelfall nur für Systemadministratoren erforderlich.

Setuid-Bit: Dieses Bit wird oft verkürzt SUID-Bit genannt. Es bewirkt, dass Programme immer so ausgeführt werden, als hätte der Besitzer selbst das Programm gestartet. Oft ist der Besitzer von Programmen *root*; dann kann jeder das Programm ausführen, als wäre er selbst *root*. Intern wird für die Ausführung des Programms die User-Identifikationsnummer des Besitzers der Datei und nicht die UID des aktuellen Benutzers verwendet.

Das Bit wird dazu eingesetzt, um gewöhnlichen Besitzern zusätzliche Rechte zu geben, die nur bei der Ausführung dieses Programms gelten. Daraus kann allerdings leicht ein Sicherheitsrisiko entstehen – insbesondere dann, wenn während der Ausführung des Programms weitere Programme gestartet werden.

Aus diesem Grund wird die Anwendung des Setuid-Bits nach Möglichkeit vermieden. Eine der wenigen Ausnahmen sind die Kommandos `mount` und `passwd`.

`ls -l` zeigt bei derartigen Programmen bei den Benutzer-Zugriffsbits den Buchstaben s (statt eines x) an. Der Oktalwert dieses Bits (für `chmod`) beträgt 4000 (s.o.).

```
root#  ls -l /bin/mount
-rwsr-xr-x    1 root   root    68508 Feb 25  01:11 /bin/mount
```

Setgid-Bit: Dieses Bit hat bei Programmen eine ähnliche Wirkung wie setuid. Allerdings wird nun während der Ausführung des Programms die Gruppen-Identifikationsnummer der Datei verwendet (und nicht die GID des aktuellen Benutzers).

`ls -l` zeigt bei derartigen Programmen für die Gruppen-Zugriffsbits den Buchstaben s (statt eines x) an. Der Oktalwert dieses Bits beträgt 2000.

```
root# ls -l /bin/mount /usr/bin/lpr
-rwsr-xr-x    1 root     root        67236 Jan 19 11:17 /bin/mount
-r-sr-sr-x    1 root     lp          16276 Mai 11 17:56 /usr/bin/lpr
```

Bei Verzeichnissen bewirkt das Setgid-Bit, dass in diesem Verzeichnis neu erzeugte Dateien der Gruppe des Verzeichnisses angehören (und nicht, wie sonst üblich, der Gruppe des Benutzers, der die Datei erzeugt). Dies ist die häufigere Anwendung.

Sticky-Bit: Dieses Bit bewirkt bei Verzeichnissen, in denen alle die Dateien ändern dürfen, dass jeder nur seine eigenen Dateien löschen darf (und nicht auch Dateien anderer Benutzer). Das Bit ist beispielsweise beim `/tmp`-Verzeichnis gesetzt. In diesem Verzeichnis darf jeder Benutzer temporäre Dateien anlegen. Es muss aber vermieden werden, dass auch jeder Benutzer nach Belieben fremde Dateien umbenennen oder löschen kann.

`ls -l` zeigt bei derartigen Programmen für alle gültigen Zugriffsbits den Buchstaben t an (statt eines x). Der Oktalwert dieses Bits beträgt 1000.

```
user$ ls -ld /tmp/
drwxrwxrwt   18 root     root         4096 Jun 14 15:34 /tmp/
```

Spezialbits in ls: Wenn Sie das Kommando `ls -l` ausführen, werden unter Umständen auch die Spezialbits S und T angezeigt. Dabei handelt es sich aber nicht um andere Spezialbits, sondern um einen Hinweis darauf, dass die Bits Setuid, Setgid oder Sticky falsch verwendet wurden:

- S: Das Setuid- oder Setgid-Bit ist gesetzt, nicht aber das Zugriffsbit x. (Setuid bzw. Setgid sind damit wirkungslos.)
- T: Das Sticky-Bit ist gesetzt, nicht aber das Zugriffsbit x für die Gruppe *others*.

3.6.3 Besitzer, Gruppe und Zugriffsbits neuer Dateien

Dieser Abschnitt beschäftigt sich mit der Frage, welche Faktoren die Zugriffsinformationen neuer Dateien bestimmen. Um das einfach auszuprobieren, verwenden Sie am besten das Kommando `touch`. Dieses Kommando erzeugt eine neue, leere Datei, falls die angegebene Datei noch nicht existiert.

Beispiel: Der Benutzer *michael* erzeugt die neue Datei `myFile1`. Es sollte nicht überraschen, dass diese Datei wieder dem Benutzer *michael* gehört – er hat sie ja gerade selbst erzeugt. Als Gruppenzugehörigkeit wurde automatisch *users* verwendet. (*users* ist die primäre Gruppe von *michael*.)

```
michael$ touch myFile1
michael$ ls -l myFile1
-rw-r--r--    1 michael   users           0 Jun 14 16:45 myFile1
```

michael gehört einer Reihe weiterer Gruppen an (Kommando *groups*). Um eine Datei zu erzeugen, die nicht der primären Gruppe angehört, muss zuerst die aktive Gruppe gewechselt werden (Kommando `newgrp`):

```
michael$  groups
users uucp dialout audio video
michael$  newgrp audio
michael$  touch myFile2
michael$  ls -l myFile2
-rw-r--r--    1 michael   audio          0 Jun 14 17:02 myFile2
```

Natürlich hätte `myFile2` auch ohne vorheriges `newgrp` erzeugt werden können. Dann hätte die Gruppenzugehörigkeit nachträglich mit `chgrp` verändert werden müssen. `newgrp` ist dann praktisch, wenn mehrere neue Dateien erzeugt werden sollen, die automatisch einer bestimmten Gruppe angehören sollen.

Besitzer und Gruppenzugehörigkeit: Aus den zwei Beispielen oben geht hervor, dass neue Dateien automatisch dem Benutzer gehören, der sie erzeugt. Als Gruppenzugehörigkeit wird normalerweise die primäre Gruppe des Benutzers verwendet (es sei denn, der Benutzer hat mit `newgrp` eine andere seiner Gruppen zur aktuellen Gruppe gemacht).

Hinweis

Wenn in einem Verzeichnis das Setgid-Bit gesetzt ist (siehe vorigen Abschnitt), dann erhalten darin erzeugte Dateien automatisch dieselbe Gruppe wie das Verzeichnis. Die aktive Gruppe des Benutzers wird nicht berücksichtigt.

Zugriffsbits: Bei den Zugriffsbits ist die Sache etwas komplizierter. Linux sieht eigentlich vor, dass neue Dateien die Zugriffsbits `rw-rw-rw` (oktal 666) bekommen, also von jedem gelesen und verändert werden dürfen. Neue Programmdateien, die von einem Compiler erzeugt werden, bekommen automatisch die Zugriffsbits `rwxrwxrwx` (777), können also auch von jedem ausgeführt werden.

Für die praktische Arbeit mit mehreren Benutzern wäre diese Default-Einstellung allerdings zu freizügig. Deswegen sehen alle Linux-Shells eine Einstellung der Zugriffsmaske (umask) vor. Dabei handelt es sich um einen Zahlenwert, der von 666 oktal (bei ausführbaren Dateien 777 oktal) abgezogen werden muss, um die Zugriffsrechte zu ergeben (echt gruselig!).

Unter Linux wird meistens ein umask-Wert von 022 verwendet. Daher bekommen neue Dateien die Zugriffsbits 666 - 022 = 644 (`rw-r--r--`), neue Programme die Zugriffsbits 777 - 022 = 755 (`rwxr-xr-x`).

Die aktuelle Einstellung des umask-Werts können Sie mit dem gleichnamigen Kommando feststellen (und auch verändern):

```
michael$  umask
022
michael$  umask 027
```

```
michael$  umask
027
```

Die Default-Einstellung des umask-Werts erfolgt in den Konfigurationsdateien der jeweiligen Shells. Für die bash (die populärste Linux-Shell) wird umask meist in `/etc/profile` oder `/etc/bashrc` eingestellt.

3.7 Linux-Verzeichnisstruktur, Device-Namen

3.7.1 Linux-Verzeichnisstruktur (Filesystem Hierarchy Standard)

Ein typisches Unix-System besteht aus Tausenden von Dateien. Während der Entwicklung von Unix haben sich bestimmte Regeln herauskristallisiert, in welchen Verzeichnissen welche Dateien normalerweise gespeichert werden. Diese Regeln wurden an die Besonderheiten von Linux angepasst und in einem eigenen Dokument zusammengefasst: dem Filesystem Hierarchy Standard (FHS). Die meisten Linux-Distributionen halten sich bis auf wenige Ausnahmen an diesen Standard. Die genaue Spezifikation des FHS finden Sie im Internet *http://www.pathname.com/fhs/*. Bei manchen Distributionen wird der Text auch im Rahmen der Dokumentation mitgeliefert.

Das Dateisystem beginnt mit dem Wurzelverzeichnis. Es enthält im Regelfall keine Dateien, sondern nur die folgenden Verzeichnisse:

- **/bin** enthält elementare Linux-Kommandos, die von allen Benutzern ausgeführt werden können. Weitere Programme befinden sich in `/usr/bin`.
- **/boot** enthält Dateien, die zum Booten des Systems (im Regelfall durch GRUB oder LILO) verwendet werden. Meistens befindet sich hier auch der Kernel.
- **/dev** enthält alle Device-Dateien. Auf alle Hardware-Komponenten wird über so genannte Devices oder Geräte-Dateien zugegriffen. Eigentlich sind es keine richtigen Dateien, sondern die Verbindung zwischen Dateischnittstelle und Hardware-Treibern).
- **/etc** enthält Konfigurationsdateien für das ganze System. Die Dateien steuern das Hochfahren des Rechners, die Tastaturbelegung, die Default-Einstellungen für diverse Linux-Komponenten (etwa für die Netzwerkkonfiguration) und für verschiedene Programme. Innerhalb von `/etc` gibt es eine Menge Unterverzeichnisse, die die Konfigurationsdateien in Gruppen ordnen.
- **/home** enthält die Heimatverzeichnisse aller Linux-Anwender.

- **/lib** enthält einige gemeinsame Bibliotheken (*shared libraries*) oder symbolische Links darauf. Die Dateien werden zur Ausführung von Programmen benötigt. `/lib/modules` enthält Kernel-Module, die im laufenden Betrieb dynamisch aktiviert bzw. deaktiviert werden.
- **/lost+found** ist normalerweise leer. Enthält es doch Dateien, dann handelt es sich um Dateifragmente, die beim Versuch, das Dateisystem zu reparieren (`fsck`), nicht mehr zugeordnet werden konnten. (Mit anderen Worten: Es wurden Sektoren gefunden, aber es ist unklar, zu welcher Datei sie einmal gehört haben.) Anstatt derartige Dateifragmente einfach zu löschen, kopiert `fsck` diese in das `lost+found`-Verzeichnis. `fsck` wird automatisch während des Systemstarts ausgeführt, wenn Linux nicht ordnungsgemäß beendet wurde (Stromausfall, Absturz etc.) oder wenn das Dateisystem längere Zeit nicht mehr überprüft wurde.
- **/mnt** bzw. **/media** enthält Unterverzeichnisse wie `cdrom` oder `floppy`, an deren Stelle externe Dateisysteme eingebunden werden. Traditionell ist hierfür `/mnt` üblich, möglicherweise wird sich in Zukunft aber `/media` durchsetzen.
- **/opt** enthält Zusatzpakete, die nachträglich installiert werden können. Auf manchen Systemen werden hierin Office-Pakete und andere kommerzielle Programme installiert. Andere mögliche Orte sind `/usr/local` und `/usr/share`.
- **/proc** enthält Unterverzeichnisse für alle laufenden Prozesse. Es handelt sich hierbei nicht um echte Dateien! Das `/proc`-Verzeichnis spiegelt lediglich die Linux-interne Verwaltung der Prozesse wider.
- **/root** enthält die Dateien des Benutzers `root`.
- **/sbin** enthält Kommandos zur Systemverwaltung. Meist können sie nur von `root` ausgeführt werden.
- **/share** enthält manchmal architekturunabhängige Dateien (also Dateien, die unabhängig vom Prozessor sind). Der korrekte Ort ist eigentlich `/usr/share`.
- **/sys** enthält voraussichtlich ab Kernel 2.6 das *sysfs*-Dateisystem. Es liefert wie das *proc*-Dateisystem Informationen über den Zustand des Rechners.
- **/tmp** enthält temporäre Dateien. Oft werden temporäre Dateien aber auch in `/var/tmp` gespeichert.
- **/usr** (unix system resources) enthält Anwendungsprogramme, das komplette X-System, die Quellcodes zu Linux etc.
- **/var** enthält veränderliche Dateien. Wichtige Unterverzeichnisse sind `adm` (distributionsabhängige Administrationsdateien), `lock` (Locking-Dateien

zum Zugriffsschutz auf Devices), `log` (Logging-Dateien), `mail` (E-Mail-Dateien, oft auch in `spool/mail`) und `spool` (zwischengespeicherte Druckdateien, News-Dateien etc.).

Die grundsätzliche Struktur der Verzeichnisse auf Wurzelebene ist also recht gut zu verstehen. Die Probleme beginnen erst mit der Unterteilung von `/usr` und `/var` in zahllose Unterverzeichnisse. Prinzipiell werden dabei viele Verzeichnisse gleich benannt wie in der Wurzel-Ebene – etwa `bin` für ausführbare Programme.

Dabei tritt das Problem auf, dass es mehrere Gruppen ausführbarer Programme gibt: textorientierte Kommandos, X-Programme etc. entsprechend viele Möglichkeiten bestehen, diese Programme zu verstecken. Aus historischen Gründen werden durch Links oft mehrere parallele Pfade verwaltet. So führt `/usr/bin/X11` zu denselben Programmen wie `/usr/X11R6/bin` (und beide Pfade sind logisch bzw. historisch begründbar).

Eine vollständige Beschreibung der Verzeichnisstruktur ist von vornherein ausgeschlossen. Dieser Abschnitt endet deswegen mit einer kurzen Beschreibung der Unterverzeichnisse von `/usr`:

/usr-Verzeichnisse	
`/usr/X11`	Link auf `/usr/X11R6`
`/usr/X11R6`	X
`/usr/bin`	ausführbare Programme
`/usr/doc`	Online-Dokumentation, FAQ (*frequently asked questions*)
`/usr/games`	Spiele; evtl. Link auf `/usr/share/games`
`/usr/include`	C-Include-Dateien
`/usr/lib`	diverse Libraries, außerdem zahllose Unterverzeichnisse für C-Compiler, diverse andere Programmiersprachen, große Programmpakete wie `emacs` oder LaTeX etc.
`/usr/local`	Anwendungen und Dateien, die nicht unmittelbar zur Linux-Distribution gehören oder später installiert wurden
`/usr/sbin`	Systemprogramme, nur von `root` ausführbar
`/usr/share`	Architekturunabhängige Daten (z. B. Emacs-Lisp-Dateien, ghostscript-Zeichensätze etc.)
`/usr/src`	Quellcode zu Linux (und evtl. zu anderen Programmen)
`/usr/tmp`	Link auf `/var/tmp` (temporäre Dateien)

3.7.2 Devices

Im Linux-Dateisystem werden nicht nur Dateien und Verzeichnisse verwaltet, sondern auch so genannte Devices. Dabei handelt es sich um speziell gekennzeichnete Dateien, in denen keine Daten gespeichert werden, sondern die vielmehr eine Verbindung zum Linux-Kernel herstellen.

Devices ermöglichen den Zugriff auf viele Hardware-Komponenten des Rechners, also etwa auf Festplatten, Diskettenlaufwerke, serielle und parallele Schnittstellen, den Arbeitsspeicher (RAM) etc. Devices sind durch drei Informationen charakterisiert: die *Major Device Number*, die *Minor Device Number* und den Typ des Zugriffs (block- oder zeichenorientiert). Die *Major Device Number* gibt an, welcher Treiber des Linux-Kernels für die Verwaltung zuständig ist. Bei vielen Treibern kann durch die *Minor Device Number* zwischen verschiedenen (verwandten) Einzelgeräten unterschieden werden, etwa beim Treiber für die seriellen Schnittstellen. Der Zugriffstyp gibt an, ob die Geräte gepuffert sind (das ist bei allen blockorientierten Geräten wie Festplatten etc. der Fall) oder nicht (zeichenorientierte Geräte wie serielle oder parallele Schnittstellen).

Wenn Sie mit `ls -l` das Inhaltsverzeichnis von `/dev` betrachten, werden statt der Dateigröße die Device-Nummern (major und minor) ausgegeben. Das erste Zeichen der Zugriffsbits lautet `b` oder `c` (block- oder zeichenorientiert). Neue Device-Dateien können mit dem Kommando `mknod` eingerichtet werden.

Manche Device-Dateien sind in Form von Links realisiert. So zeigt `/dev/mouse` auf die Device-Datei, die für die Schnittstelle zuständig ist, an der die Maus tatsächlich angeschlossen ist (häufig die erste serielle Schnittstelle oder `psaux` bei PS/2-Mäusen).

Um gezielt steuern zu können, welcher Benutzer auf welche Devices zugreifen darf, sind den Devices unterschiedliche Benutzergruppen zugeordnet. Beispielsweise sind die Devices `/dev/ttyS*` für die seriellen Schnittstellen üblicherweise der Gruppe `uucp` zugeordnet:

```
root#  ls -l /dev/ttyS1
crw-rw----   1 root     uucp       5,  65 Jul 18  /dev/ttyS1
```

Wenn der Systemadministrator möchte, dass die Userin *gundel* über die serielle Schnittstelle direkt auf ein Modem zugreifen darf, fügt er *gundel* zur Gruppe *uucp* hinzu (mit dem Kommando `usermod -G` oder durch Bearbeiten der Datei `/etc/group`).

Einige Device-Dateien haben eine besondere Funktion: So dient `/dev/null` als „schwarzes Loch“, an das Daten gesendet werden können, die dort für immer verschwinden (etwa zur Umleitung von Kommando-Ausgaben, die nicht angezeigt werden sollen). `/dev/zero` ist eine unerschöpfliche Quelle von 0-Bytes, die manchmal dazu verwendet wird, Dateien bis zu einer vorgegebenen Größe mit Nullen zu füllen.

Eine vollständige Beschreibung aller unter Linux zurzeit definierten Devices samt der dazugehörigen Device-Nummern finden Sie in der Datei `/usr/src/linux/Documentation/devices.txt` (nur, wenn der Kernel-Quellcode installiert ist).

Die Zukunft des Device-Systems: Das gegenwärtige System von Device-Dateien ist aus zwei Gründen unflexibel und unpraktisch:

- Um jede nur denkbare Hardware des Benutzers zu unterstützen, erzeugen die meisten Distributionen bei der Installation eine riesige Anzahl von Device-Dateien. (Bei Red Hat 9 gab es beispielsweise beinahe 8000 derartige Dateien!) Tatsächlich genutzt werden auf jedem Rechner aber höchstens ein paar Prozent der Dateien.
- Aus historischen Gründen handelt es sich bei der Device-Nummer bis einschließlich Kernel 2.4 um eine 16-Bit-Zahl. Da jedes Jahr neue Hardware-Komponenten hinzukommen, wird der verfügbare Zahlenraum für neue Devices immer enger.

Die größte Neuerung in Kernel 2.6 ist die Vergrößerung der Device-Nummern auf 64 Bits. Bereits vorhandene Device-Nummern sollten sich dadurch nicht ändern; ganz sind Kompatibilitätsprobleme aber nicht auszuschließen. Offen ist noch, ob sich am prinzipiellen Umgang mit Device-Dateien in Zukunft etwas ändern wird.

Was ist `udev`?

Salopp gesagt erstellt und entfernt es Einträge in `/dev`, basierend auf der aktuellen Systemkonfiguration. Prinzipiell ist `udev` ein modulares System zur automatischen Erstellung von Gerätedateien in `/dev`.

Im Verzeichnis `/dev` befinden sich normalerweise Gerätedateien (Device Nodes), die, wie oben beschrieben, den Zugriff auf Geräte (z. B. Festplatten, Maus, Soundkarte) erlauben. Diese Gerätedateien werden normalerweise mit `MAKEDEV` angelegt, denn ohne Gerätedatei ist kein Zugriff auf ein Gerät möglich. Aus diesem Grunde existieren normalerweise Unmengen von Gerätedateien für größtenteils nicht vorhandene Geräte. Das `udev`-Dateisystem legt diese Gerätedateien dynamisch an, der Einsatz von `MAKEDEV` ist nicht mehr nötig und es existieren nur noch Gerätedateien für existierende Geräte mit Treiber. Mit andren Worten: nie wieder nach nicht existierenden Geräten suchen oder Device Nodes anlegen müssen.

Dies wurde erreicht durch Überwachen der vom Programm `hotplug` generierten Ereignisse im System und Auslesen von Informationen zu diesen Ereignissen aus dem Dateisystem `sysfs`. `udev` arbeitet unter Benutzung von `hotplug`-Aufrufen des Kernels, wann immer ein Gerät zum Kernel hinzugefügt oder daraus entfernt wird. Die Namensgebung und Zugangsberechtigungen werden im User-Space ausgeführt. Die Ziele des `udev` Projekts sind:

- Läuft im User-space
- Erstellt/entfernt dynamisch Gerätedateien
- Liefert konsequente Benennung
- Liefert ein User-space API

Um diese Funktionen zu liefern, wird `udev` in drei unterschiedlichen Projekten entwickelt: `namedev`, `libsysfs` und natürlich `udev`.

- `namedev`
  `namedev` erlaubt es, Geräte unabhängig vom `udev`-Programm zu benennen. Dies ermöglicht flexible Benennungsrichtlinien und Namensschemata. Das Subsystem zur Gerätebenennung liefert ein Standardinterface, das `udev` benutzen kann. Derzeit gibt es nur ein einzelnes Benennungsschema von `namedev`, LANANA, das auch von der Mehrheit der Linux-Systeme verwendet wird. `namedev` verwendet eine fünfstufige Prozedur, um den Namen eines bestimmten Gerätes herauszufinden. Wenn in einem dieser Schritte der Gerätename gefunden wird, verwendet es diesen Namen. Diese Schritte sind:

  1. Beschriftung oder Seriennummer
  2. Bus-Gerätenummer
  3. Bus-Topologie
  4. Statisch vergebener Name
  5. Vom Kernel gelieferter Name

  Der erste Schritt überprüft, ob das Gerät ein einzigartiges Identifikationsmerkmal hat. Zum Beispiel haben USB-Geräte eine einzigartige USB-Seriennummer und SCSI-Geräte eine einzigartige UUID. Wenn `namedev` eine Übereinstimmung zwischen dieser einzigartigen Nummer und einer gegebenen Konfigurationsdatei findet, dann wird der von der Konfigurationsdatei gelieferte Name verwendet.

  Der zweite Schritt überprüft die Bus-Gerätenummer. Für nicht-hot-swappable Umgebungen ist diese Prozedur ausreichend, um ein Hardware-Gerät zu identifizieren. Zum Beispiel verändern sich PCI-Busnummern selten in der Lebenszeit eines Systems. Findet `namedev` eine Übereinstimmung mit dieser Position und einer gegebenen Konfigurationsdatei, wird der von der Konfigurationsdatei gelieferte Name verwendet.

  Auch die Bus-Topologie ist ein eher statischer Weg zur Definition von Geräten. Wenn die Position des Gerätes zu einer vom Benutzer gelieferten Einstellung passt, wird der beiliegende Name verwendet.

  Der vierte Schritt ist eine einfache Stringersetzung. Wenn der Kernel-Name (der Standardname) des Device zu einem gegebenen Ersatzstring passt, wird der Ersatzname stattdessen verwendet.

  Der letzte Schritt nimmt den vom Kernel gelieferten Standardnamen. In den meisten Fällen ist dies ausreichend, da es zur augenblicklich üblichen Gerätebenennung passt.

- `libsysfs`
  `udev` interagiert mit dem Kernel durch das `sysfs`-Pseudodateisystem. Das

libsysfs-Projekt liefert ein Standard-API, um auf die durch das sysfs-Dateisystem gegebenen Informationen zuzugreifen. Dies erlaubt eine Abfrage aller Art von Hardware, ohne dass man Vermutungen über die Art der Hardware anstellen muss.

- `udev`
  Jedes Mal, wenn der Kernel ein Update in der Gerätestruktur feststellt, ruft er das Programm `hotplug` auf. Hotplug führt die Anwendung aus, welche im Verzeichnis `/etc/hotplug.d/default` verlinkt ist. Hotplug übergibt die Informationen vom Kernel an das `udev`-Programm, das die notwendigen Aktionen (Erstellen oder Entfernen von Gerätedateien) in `/dev` ausführt.

Hinweis

Bei manchen Distributionen werden bei einem Login die Zugriffsrechte wichtiger Device-Dateien so eingestellt, dass der aktuelle Benutzer die entsprechenden Geräte uneingeschränkt nutzen kann. Das hat freilich die Konsequenz, dass sich die Zugriffsrechte wie von Zauberhand immer wieder verändern! Red Hat bzw. Fedora greift zu diesem Zweck auf PAM zurück, Suse setzt stattdessen den Dämon `resmgrd` ein.

3.7.3 Dateitypen (MIME)

Sie klicken in einem Webbrowser oder Dateimanager auf einen Link, der auf eine MP3-Datei verweist – und statt dass automatisch ein MP3-Player startet, passiert nichts! Schuld daran ist in den meisten Fällen MIME.

MIME steht für *Multipurpose Internet Mail Extensions*. Ursprünglich bezog sich MIME auf E-Mail-Attachments. Wenn mit einer E-Mail beispielsweise eine PostScript- oder GIF-Datei mitgesandt wird, dann sollte der E-Mail-Client wissen, mit welchem Programm diese Datei betrachtet bzw. bearbeitet werden kann. Damit das funktioniert, ist die MIME-Konfiguration erforderlich.

Mittlerweile reicht die Anwendung von MIME aber viel weiter: Wenn Sie im Dateimanager oder Webbrowser einen Link auf eine Datendatei verfolgen, sollte auch dieses Programm wissen, wie es mit diesen Daten umgehen soll. Die Bedeutung einer korrekten MIME-Konfiguration erstreckt sich also nicht nur auf E-Mails, sondern auf alle Programme, die mit unterschiedlichen Datentypen zurechtkommen müssen.

Linux wäre nicht Linux (oder Unix), wenn es einen zentralen Ort für die MIME-Konfiguration gäbe. Stattdessen gibt es eine ganze Menge. Die MIME-Daten für KDE-Programme, Gnome-Programme, diverse Webbrowser, für das Drucksystem CUPS etc. werden jeweils separat verwaltet. Außerdem gibt es noch eine zentrale MIME-Konfiguration für alle Programme, die keine eigenen MIME-Konfigurationsdateien verwalten.

Die Aufteilung der MIME-Konfiguration auf mehrere Orte hat aber durchaus ihre Gründe. Sowohl KDE als auch Gnome verwenden ein Konzept, das Komponenten zur Bearbeitung verschiedener Datentypen vorsieht. Wenn im KDE-Dateimanager eine PNG-Bilddatei angezeigt werden soll, wird einfach die entsprechende Komponente geladen und ausgeführt. Da die KDE- und Gnome-Bibliotheken zueinander weitgehend inkompatibel sind, wäre es fatal, wenn der KDE-Dateimanager versuchen würde, eine Gnome-Komponente auszuführen (oder umgekehrt). Um das zu vermeiden, verwenden KDE und Gnome jeweils ihre eigene MIME-Datenbank. Ähnlich ist die Argumentation auch bei allen anderen Programmen mit eigener MIME-Konfiguration.

Bei vielen MIME-Konfigurationsdateien muss darüber hinaus zwischen der globalen und der individuellen Konfiguration unterschieden werden, also zwischen der Default-Einstellung für alle Anwender und benutzerspezifischen Einstellungen.

Allgemeine MIME-Konfiguration: Die allgemeinen MIME-Konfigurationsdateien werden nur von den Programmen berücksichtigt, die keine eigenen MIME-Dateien verwalten. Die Einstellungen sind auf zwei Dateien verteilt, von denen es jeweils eine globale und eine benutzerspezifische Version gibt:

```
/etc/mime.types        global: Dateitypen
/etc/mailcap           global: Programme
~/.mime.types          lokal:  Dateitypen
~/.mailcap             lokal:  Programme
```

`mime.types` enthält eine Liste, die die Zuordnung zwischen Dateitypen (erste Spalte) und Dateikennungen (alle weiteren Spalten) herstellt. Die erste Beispielzeile ordnet dem Typ *application/pdf* die Kennung `*.pdf` zu. In `mime.types` wird zum Teil zwischen Text- und X-Applikationen unterschieden, weswegen Sie Dateitypen wie *application/x-name* finden werden.

`mailcap` gibt an, welches Programm zur Anzeige bzw. Bearbeitung eines bestimmten Dateityps verwendet werden soll. Die folgende Zeile besagt, dass zur Anzeige von PDF-Dateien das Programm `xpdf` verwendet werden soll. Im Gegensatz zu `mime.types` müssen die Spalten in `mailcap` durch Semikola getrennt werden. `%s` ist ein Platzhalter für den Dateinamen.

```
# in /etc/mailcap
application/pdf; xpdf %s
```

3.7.4 Magic-Dateien

MIME ist für die Zuordnung zwischen dem Dateityp und den dazu passenden Programmen zuständig. Aber wie wird der Dateityp überhaupt festgestellt? Der Normalfall besteht darin, dass die Dateiendung den Dateityp angibt. Die Dateiendung `*.ps` deutet also beispielsweise auf eine PostScript-Datei hin. Natürlich

ist das keine besonders elegante und sichere Methode der Dateikennung. Darum bietet Linux noch eine bessere Methode. Fast jede Datei kann anhand ihres Inhalts identifiziert werden (so beginnt beispielsweise ein GIF-Bild mit der Buchstabenfolge „GIF89“ bzw. „GIF87“).

Das Programm `file` oder entsprechende KDE- oder Gnome-Äquivalente erkennen den Dateityp aus dem Inhalt der ersten Bytes bzw. aus in der Datei enthaltenen charakteristischen Zeichenketten. Das Erkennungsverfahren basiert auf einer so genannten Magic-Datei, die Informationen darüber enthält, welche Byte- und Zeichenmuster eine Datei enthalten kann (und die man, wie bei Linux üblich, selbst erweitern kann).

Wie nicht anders zu erwarten, gibt es auch für die Magic-Dateien unterschiedliche Speicherorte, die noch dazu distributionsabhängig sind. Bei Red Hat bzw. Fedora gibt es die folgenden Magic-Dateien. (In Klammern sind jeweils die zugrunde liegenden Pakete angegeben.)

`/usr/share/magic`	*allgemeine Magic-Datei (file)*
`/etc/mime-magic`	*Gnome-Magic-Datei (gnome-libs)*
`/etc/gnome-vfs-mime-magic`	*Gnome-2-Datei (gnome-mime-data)*
`/usr/share/mimelnk/magic`	*KDE-Magic-Datei (kdelibs)*

Bei Suse gibt es diese Magic-Dateien:

`/etc/magic`	*Link auf* `/usr/share/misc/magic`
`/usr/share/misc/magic`	*allgemeine Magic-Datei (file)*
`(kdedir)/share/mimelnk/magic`	*KDE-Magic-Datei (kdelibs)*
`/etc/opt/gnome/mime-magic`	*Gnome-Magic-Datei (gnome-libs)*
`/etc/opt/gnome2/gnome-vfs-mime-magic`	*Gnome-2-Datei (gnome-mime-data)*

3.8 CD-ROMs, DVDs, Disketten und Windows-Partitionen

Etwas ungewohnt für Windows-Anwender ist die Art und Weise, wie unter Linux auf CD-ROMs, DVDs, Disketten, externe Datenträger inklusive USB-Sticks sowie auf Windows-Partitionen zugegriffen wird. Anders als unter Windows gibt es dafür keine Buchstaben. Stattdessen erfolgt der Zugriff direkt über ein Verzeichnis des Dateisystems. Der Inhalt einer CD-ROM steht dann beispielsweise unter dem Verzeichnis `/media/cdrom` oder `/mnt/cdrom` zur Verfügung.

Der zweite Unterschied besteht darin, dass externe Datenträger bzw. fremde Partitionen explizit in das Dateisystem eingebunden und wieder abgemeldet werden müssen. Falls Sie in einer Textkonsole arbeiten, müssen Sie zum Einbinden des Datenträgers das Kommando `mount` ausführen (zum Abmelden `umount` – ja, ohne „n“!). Gebunden wird ein externer Datenträger immer auf ein vorhandenes Verzeichnis. Bei den meisten Distributionen sind solche Verzeichnisse bereits vorhanden (unter `/media` oder `/mnt`. Das Kommando benötigt in der

Regel die Angabe eines Verzeichnisses und eines Gerätes, manchmal noch den Typ des Dateisystems, zum Beispiel zum Einbinden einer Windows-Diskette:

```
user$  mount -t vfat /mnt/floppy /dev/fd0
```

CD-ROMs oder andere Datenträger können Sie erst wieder aus dem Laufwerk entfernen, nachdem die CD-ROM wieder vom Dateisystem getrennt wurde (`umount`). Dieses Kommando kann allerdings erst dann ausgeführt werden, wenn kein Programm mehr auf Daten der CD-ROM zugreift. Als Zugriff gilt auch, wenn in irgendeinem Programm (Dateimanager, Shell-Fenster, Konsole) das aktuelle Verzeichnis ein CD-ROM-Verzeichnis ist.

3.8.1 Automatischer Zugriff auf Datenträger unter KDE und Gnome

Im Idealfall funktioniert der Zugriff auf diverse Datenträger (CD-ROMs, DVDs, externe Festplatten, USB-Sticks, Disketten) weitgehend automatisch. Für CD-ROM- bzw. DVD-Laufwerke sollten auf dem KDE- bzw. Gnome-Desktop entsprechende Symbole erscheinen, die Sie nur anklicken müssen, wenn Sie Daten lesen möchten. Bei USB- und Firewire-Datenträgern sollte ein derartiges Icon automatisch erscheinen, sobald Sie das Gerät einstecken.

Vereinzelt verwenden Distributionen auch so genannte Supermount- oder Automount-Verfahren. Dabei verhält sich das System eher wie unter Windows gewohnt: Sobald Sie eine CD oder DVD einlegen, wird sie automatisch in das Dateisystem eingebunden. Wenn Sie die CD aus dem Laufwerk entfernen, wird sie automatisch wieder aus dem Dateisystem entfernt.

Bevor Sie ein externes Gerät wieder abstecken bzw. eine Diskette wechseln, sollten Sie den Datenträger explizit abmelden. Bei vielen Distributionen gilt diese Regel auch für unveränderliche Medien wie CDs und DVDs. Dazu klicken Sie das Icon mit der rechten Maustaste an und führen ein Kommando der Art *Laufwerkeinbindung lösen* aus.

Leider hinkt die Realität hinter diesem Idealfall ein Stück hinterher. So wie es hier beschrieben wurde, funktioniert es nur manchmal, d. h. je nach Distribution nur unter KDE oder Gnome, je nach Distribution nur für CDs/DVDs, aber nicht für externe Datenträger etc. Generell werden die Linux-Distributionen immer besser, aber bis zur Perfektion hat es noch keine Distribution gebracht. Wenn der Zugriff nicht automatisch funktioniert, müssen Sie mit ein paar Kommandos selbst nachhelfen.

3.8.2 Manueller Zugriff auf CD-ROMs

Wenn Sie in einer Textkonsole arbeiten bzw. wenn kein entsprechend konfigurierter Desktop zur Verfügung steht, müssen Sie `mount` und `umount` selbst durchführen. Im einfachsten Fall lauten die Kommandos:

```
root# mount /mnt/cdrom
root# umount /mnt/cdrom
```

Bei manchen Distributionen müssen Sie statt `/mnt/cdrom` ein anderes Verzeichnis angeben – z. B. `/media/cdrom` oder `/media/dvd` (Suse). Das erforderliche Verzeichnis können Sie feststellen, wenn Sie sich die Datei `/etc/fstab` ansehen. Dort sind alle bekannten Laufwerke und Partitionen mit dem unter Linux gültigen Verzeichnis, dem Typ des Dateisystems und dem anzusprechenden Gerät aufgezählt.

Je nach Konfiguration dürfen die Kommandos `mount` und `umount` aus Sicherheitsgründen nur von `root` ausgeführt werden.

Lange Dateinamen auf CD-ROMs

Fast alle Linux-CDs verwenden lange Dateinamen, die im ISO-9660-Standard eigentlich nicht vorgesehen waren. Um solche Dateinamen dennoch zu unterstützen, verwenden Linux-CDs die so genannte Rockridge-Extension. Diese Erweiterung zum ISO-Standard gibt an, wie Zusatzinformationen zu den Dateien gespeichert werden können. Die Rockridge-Extension ist keine Linux-spezifische Besonderheit, sondern ein Standard in der Unix-Welt.

Die Rockridge-Extension ist freilich nicht die einzige Möglichkeit, lange Dateinamen auf CDs unterzubringen. Apple und Microsoft (Joliet) haben jeweils eigene ISO-9660-Erweiterungen definiert.

Linux ist in der Lage, lange Dateinamen auch gemäß der Joliet-Erweiterung korrekt zu interpretieren. Damit können Windows-CDs unter Linux problemlos genutzt werden. Umgekehrt gilt das leider nicht, d. h. Windows ist nicht in der Lage, lange Dateinamen gemäß der Rockridge-Spezifikation anzuzeigen. Stattdessen werden bei solchen CDs unter Windows alle Dateinamen auf 8+3 Zeichen verstümmelt.

3.8.3 Manueller Zugriff auf Disketten

Im Prinzip gilt dieselbe Vorgehensweise auch für Disketten. Statt `/mnt/cdrom` verwenden Sie nun `/mnt/floppy` oder `/media/floppy`:

```
root# mount /mnt/floppy
root# umount /mnt/floppy
```

Wenn es nur darum geht, eine einzelne Datei auf eine Diskette mit Windows-Formatierung zu schreiben bzw. von dort zu lesen, können Sie sich die Mühe mit `mount` und `umount` auch sparen. Stattdessen können Sie einfach die Kommandos `mdir`, `mcopy` etc. verwenden. Das Inhaltsverzeichnis einer Diskette können Sie beispielsweise so ermitteln:

```
root# mdir A:
```

Beachten Sie bitte, dass diese Kommandos nicht bei jeder Distribution automatisch installiert sind und dass sie nicht für Linux-Disketten verwendet werden können (also nicht für Disketten mit `ext2`-Dateiformat).

3.8.4 Manueller Zugriff auf Windows-Partitionen

Die Verzeichnisse zum Lesen/Schreiben auf Windows-Partitionen sind ebenfalls bei jeder Distribution anders (siehe `/etc/fstab`!): Red Hat und Fedora sehen per Default keinen Zugriff auf Windows-Partitionen vor, Suse bietet die Verzeichnisse `/windows/C` und `/windows/D`.

Soweit diese Verzeichnisse nicht bereits in das Dateisystem integriert sind, müssen Sie auch hier `mount` ausführen. Per Default ist das oft aus Geschwindigkeits- und Sicherheitsgründen nicht der Fall. Generell können Sie Daten in Windows-9x/ME-Partitionen lesen und verändern, Daten in Windows-NT/2000/XP-Partitionen dagegen nur lesen.

Mit dem Kommando `df` können Sie feststellen, welche Laufwerke und Partitionen momentan in das Dateisystem eingebunden sind und wie viel Speicher dort noch frei ist. Im Beispiel unten sind außer dem Startverzeichnis `/` noch ein CD-ROM-Laufwerk und eine halb volle Windows-Partition in das Dateisystem eingebunden:

```
root# df
Filesystem   1k-blocks      Used Available Use% Mounted on
/dev/hdb8      2521936   1545660    848160  65% /
/dev/hdd       7989184   7989184         0 100% /media/cdrom
/dev/hda5      4096540   1943876   2152664  48% /windows/D
```

3.8.5 USB-, Firewire- und PCMCIA-Laufwerke, USB-Memory-Sticks

Diese Datenträger werden im laufenden Betrieb mit dem Computer verbunden und auch wieder abgesteckt. Intern werden die Laufwerke meist wie SCSI-Laufwerke behandelt; einzig ein getesteter PCMCIA-Flash-Adapter machte sich als neues IDE-Laufwerk bemerkbar.

Die meisten Distributionen laden automatisch die erforderlichen USB-, Firewire- und SCSI-Kernel-Module, um mit dem Laufwerk zu kommunizieren. Im Idealfall

erscheint unter KDE oder Gnome ein entsprechendes Icon oder ein Dateimanager mit dem Inhaltsverzeichnis des Datenträgers. Allerdings können gewöhnliche Benutzer in der Regel nur Daten lesen; nur *root* darf die Daten auch verändern.

Datenträger manuell einbinden: Nach dem Einstecken müssen Sie feststellen, welchen internen Device-Namen das Gerät bekommen hat. Wenn es sich um das einzige externe bzw. SCSI-Gerät handelt, lautet der Name immer `/dev/sda`. Die Daten befinden sich zumeist auf der ersten Partition des Datenträgers mit dem Namen `/dev/sda1`. Falls es mehrere externe Geräte gibt, wird aus `sda` entsprechend `sdb`, `sdc` etc.

Einen Überblick über alle Datenträger (inklusive Festplatten, aber ohne CD- und DVD-Laufwerke) gibt das Kommando `fdisk -l`. Beim folgenden Beispiel ist `/dev/hda` die eingebaute Festplatte mit diversen Linux- und Windows-Partitionen, und `/dev/sda` ist eine externe USB-Festplatte:

```
root# fdisk -l
Disk /dev/sda: 203.9 GB, 203927060480 bytes
255 heads, 63 sectors/track, 24792 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
   Device Boot    Start       End    Blocks   Id  System
/dev/sda1             1     15298 122881153+   7  FAT32

Disk /dev/hda: 123.5 GB, 123522416640 bytes
255 heads, 63 sectors/track, 15017 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
...
```

USB-Memory-Sticks können auch wie eine so genannte Superfloppy formatiert sein. Das bedeutet, dass es keine Partitionierungstabelle gibt. In diesem Fall wird das gesamte Laufwerk über den Device-Namen `/dev/sda` angesprochen (anstatt wie sonst üblich mit `/dev/sda1` die Nummer einer bestimmten Partition anzugeben).

Wissen Sie die Device-Nummer, erstellen Sie ein neues Verzeichnis und führen das folgende `mount`-Kommando aus:

```
root# mkdir /usbstick
root# mount /dev/sda1 /usbstick
```

Der Dateisystemtyp auf externen Datenträgern ist beliebig. In der Praxis kommt bei externen Festplatten und USB-Memory-Sticks aber am häufigsten VFAT (Windows) zum Einsatz. Das gilt auch für Speicherkarten für diverse elektronische Geräte (PDAs, Digitalkameras etc.), die mittels eines USB- oder PCMCIA-Adapters genutzt werden. Selbstverständlich können Sie den Datenträger unter Linux mit einem beliebigen Dateisystem neu formatieren bzw. die Partitionierung des Datenträgers verändern (z. B. mit `fdisk`).

Haben Sie alle Daten gelesen oder geschrieben, führen Sie wie üblich `umount` aus. Entfernen Sie auf keinen Fall die USB- oder Firewire-Verkabelung, bevor Sie `umount` ausführen – Sie riskieren Datenverluste!

```
root#  umount /usbstick
```

fstab-Eintrag: Ein grundsätzliches Problem besteht darin, dass nur `root` das Kommando `mount` ausführen darf. Die übliche Lösung besteht darin, einfach in `fstab` eine entsprechende Zeile mit der Option *user* einzufügen. Für einen USB-Memory-Stick mit VFAT-Dateisystem könnte diese Zeile so aussehen:

```
# /etc/fstab: USB-Stick
/dev/sda1  /usbstick  vfat  user,umask=0,noauto  0 0
```

Damit kann jeder Benutzer den USB-Stick mit `mount /usbstick` in das Dateisystem einbinden und die enthaltenen Daten lesen und verändern. Die Vorgehensweise hat allerdings zwei gravierende Nachteile:

- Wenn statt des USB-Sticks eine externe Festplatte eingesteckt wird, hat jeder Benutzer Lese- und Schreibzugriff auf die erste Partition dieser Festplatte (sofern sich dort ein VFAT-Dateisystem befindet). Das ist aber aus Sicherheitsgründen meist nicht beabsichtigt.

  Das Sicherheitsrisiko lässt sich ein wenig reduzieren, wenn eine Gruppe *extdisk* eingerichtet wird und in `fstab` die Optionen *gid* und *umode* so eingestellt werden, dass nur *extdisk*-Gruppenmitglieder Schreibrechte bekommen. Wenn die Gruppen-ID 1234 lautet, dann bewirkt *gid=1234,umode=002*, dass *root* sowie alle *extdisk*-Mitglieder uneingeschränkte Rechte haben. Andere Anwender dürfen die Dateien und Verzeichnisse nur lesen.

- Wenn es mehrere externe Geräte gibt, ändert sich der Device-Name, je nachdem, in welcher Reihenfolge die Geräte eingesteckt werden. Wenn der USB-Stick als zweites oder drittes Gerät eingesteckt wird, scheitert ein Zugriff über das Verzeichnis `/usbstick`.

3.8.6 Aktuellen Zustand des Dateisystems ermitteln

Wenn Sie wissen möchten, wie Ihr Linux-System zurzeit organisiert ist, führen Sie am einfachsten das Kommando `df` aus. Dieses Kommando zeigt an, an welcher Stelle im Dateisystem Festplatten, Datenträger etc. eingebunden sind und wie viel Platz auf den einzelnen Festplatten noch frei ist. Das Kommando `du` zeigt für alle Dateien eines Verzeichnisses (und dessen Unterverzeichnisse) an, wie groß die Dateien sind. Mit dem Parameter `-s` erhält man nur die Summe der Dateigrößen.

Die Datei /etc/mtab enthält noch detailliertere Informationen über die eingebundenen Dateisysteme. In ihr sind auch alle virtuellen und Netzwerk-Dateisysteme enthalten. Außerdem enthält /etc/mtab alle bei mount verwendeten Optionen. Das Format von /etc/mtab ist identisch mit dem von /etc/fstab.

```
root#  cat /etc/mtab
/dev/hda12 /             ext3     rw 0 0
/dev/hda11 /kofler       ext3     rw 0 0
/dev/hda13 /fedora       ext3     rw 0 0
mars:/data /data         nfs      rw,noexec,nosuid,nodev,
                                    addr=192.168.0.10 0 0
/dev/sr0   /media/dvdrec iso9660  ro,nosuid,nodev 0 0

tmpfs      /dev/shm      tmpfs    rw 0 0
proc       /proc         proc     rw 0 0
```

3.8.7 /etc/fstab – Dateisysteme automatisch einbinden

Es wäre sehr mühsam, wenn Sie nach jedem Systemstart diverse Partitionen neu einbinden müssten, wenn Sie bei jedem CD-Wechsel mount mit allen Optionen angeben müssten etc. Der Schlüssel zur Arbeitserleichterung heißt /etc/fstab.

Diese Datei gibt an, welche Datenträger beim Systemstart in das Dateisystem aufgenommen werden. Auf jeden Fall muss fstab die Systempartition sowie alle zur internen Verwaltung notwendigen Dateisysteme enthalten. Je nach Distribution kann eine minimale fstab-Datei wie folgt aussehen:

```
# zwei beispielhafte Zeilen in /etc/fstab
/dev/hda2   /       ext2    defaults 1 1
none        /proc   proc    defaults 0 0
... usw.
```

Durch die erste Zeile wird die zweite Festplattenpartition der ersten IDE-Platte als Systemverzeichnis genutzt. Je nachdem, auf welcher Festplattenpartition Sie Linux installiert haben, müssen Sie statt hda2 natürlich den Device-Namen Ihrer Linux-Partition angeben!

Mit der zweiten Zeile wird das System zur Prozessverwaltung in das Dateisystem eingebunden. Die Dateien und Verzeichnisse des /proc-Verzeichnisses existieren nicht tatsächlich auf der Festplatte; es handelt sich nur um ein Abbild von Daten, die kernel-intern verwaltet werden.

Die Datei /etc/fstab kann sich im laufenden Betrieb ändern! Manche Distributionen überwachen durch einen Hardware- oder Hotplug-Dämon den USB- und Firewire-Bus bzw. die PCMCIA-Slots eines Notebooks. Sobald neue Datenträger angesteckt werden, fügt das Hintergrundprogramm eine entsprechende Zeile in

`/etc/fstab` ein. Wenn der Datenträger entfernt wird, wird eventuell auch die entsprechende Zeile in `/etc/fstab` entfernt.

Diese automatischen Veränderungen an `/etc/fstab` können lästig sein, wenn dadurch eigene Einträge überschrieben oder gelöscht werden. Red Hat bzw. Fedora kennzeichnet dynamische Einträge immerhin durch die zusätzliche Option *kudzu*. Bei Suse fehlt eine entsprechende Markierung.

Aus dem obigen Beispiel geht bereits das prinzipielle Format von `fstab` hervor: Jede Zeile beschreibt in sechs Spalten ein Dateisystem (eine Partition, einen Datenträger).

In der **ersten Spalte** wird der Device-Name des Datenträgers angegeben. Statt des Device-Namens können Sie auch den Partitionsnamen (*filesystem volume name* oder die ID-Nummer der Partition angeben. Die korrekte Syntax lautet in diesem Fall *LABEL=zeichenkette* oder *UUID=nnn-nnn*. Je nach Dateisystemtyp gibt es eigene Werkzeuge, um den Partitionsnamen bzw. die UUID zu lesen oder zu verändern.

Der Vorteil von Partitionsnamen im Vergleich zu Device-Namen besteht darin, dass die Angabe selbst dann noch korrekt ist, wenn sich der Device-Name geändert hat. Das kann z. B. bei SCSI-Festplatten relativ leicht passieren, wenn ein zusätzliches Laufwerk eingefügt wird. Aus `/dev/sdb3` kann dann durchaus `/dev/sdc3` werden.

Die **zweite Spalte** gibt an, bei welchem Verzeichnis der Datenträger in den Dateibaum eingebunden wird. Die in der zweiten Spalte angegebenen Verzeichnisse müssen bereits existieren. Die Verzeichnisse müssen nicht leer sein, allerdings können Sie nach dem Einbinden des Dateisystems auf die darin enthaltenen Dateien nicht mehr zugreifen (sondern nur auf die Dateien des eingebundenen Datenträgers).

Die **dritte Spalte** gibt das Dateisystem an. Die folgende Tabelle listet in alphabetischer Reihenfolge die wichtigsten Dateisysteme auf:

`auto`	Dateisystem automatisch erkennen (CD-ROMs, Disketten)
`devpts`	Pseudoterminals gemäß Unix-98-Spezifikation
`ext2`	Linux-Standard (ext2-Dateisystem)
`ext3`	ext3-Dateisystem
`iso9660`	CD-ROMs, DVDs
`nfs`	Unix-Netzwerkverzeichnis (NFS)
`ntfs`	Windows-NT/2000/XP-Dateisystem (nur Lesezugriff)
`proc`	Prozessverwaltung (`/proc`)
`reiserfs`	Reiser-Dateisystem
`smbfs`	Windows-Netzwerkverzeichnis (Samba)
`swap`	Swap-Partitionen oder -Dateien
`sysfs`	Systemverwaltung (`/sys`, Kernel 2.6)
`udf`	Universal Disk Format (DVDs, CD-RWs)
`usbdevfs`	Verwaltung von USB-Geräten (Kernel 2.4)

`usbfs`	Verwaltung von USB-Geräten (Kernel 2.6)
`vfat`	Windows-9x/ME-Dateisystem

Es ist auch zulässig, mehrere Dateisysteme gleichzeitig anzugeben (durch Kommata getrennt). Beispielsweise bietet sich `iso9660,udf` für CD- und DVD-Laufwerke an, weil für CDs und DVDs in der Regel nur diese beiden Dateisysteme in Frage kommen. `mount` entscheidet sich zwischen den zur Auswahl stehenden Systemen automatisch für das richtige.

Die **vierte Spalte** bestimmt Optionen für den Zugriff auf den Datenträger. Mehrere Optionen werden durch Kommata getrennt. Dabei dürfen keine Leerzeichen eingefügt werden! Die folgende Tabelle zählt die wichtigsten universellen `mount`-Optionen auf.

`defaults`	Default-Optionen verwenden
`dev`	Kennzeichnung von Character- oder Block-Devices auswerten
`exec`	Programmausführung zulassen (für CD-ROM-Laufwerke)
`kudzu`	Red-Hat- bzw. Fedora-spezifisch, siehe unten
`noauto`	Datenträger nicht beim Systemstart einbinden
`nodev`	Kennzeichnung von Character- oder Block-Devices ignorieren
`noexec`	Keine Programmausführung erlaubt
`nosuid`	sid- und gid-Zugriffsbits nicht auswerten
`ro`	Read only (Schreibschutz)
`sw`	Swap (Swap-Datei oder -Partition)
`suid`	sid- und gid-Zugriffsbits auswerten
`sync`	Schreibzugriffe nicht puffern (sicherer, aber langsamer)
`owner`	Der Besitzer darf `(u)mount` ausführen
`user`	Jeder darf `mount` ausführen, aber nur der Benutzer des letzten `mount`-Aufrufs darf `umount` ausführen
`users`	Jeder darf `(u)mount` ausführen

Die **fünfte Spalte** enthält Informationen für das Programm `dump` und wird zurzeit ignoriert. Es ist üblich, für die Systempartition 1 und für alle anderen Partitionen oder Datenträger 0 einzutragen.

Die **sechste Spalte** gibt an, ob und in welcher Reihenfolge die Dateisysteme beim Systemstart überprüft werden sollen. Bei den meisten Distributionen wird 1 für die Systempartition und 0 für alle anderen Partitionen eingetragen. Das bedeutet, dass beim Rechnerstart nur die Systempartition auf Fehler überprüft und gegebenenfalls repariert wird.

Falls Sie möchten, dass weitere Partitionen automatisch überprüft werden, geben Sie bei diesen Partitionen die Ziffer 2 an. Bei allen Dateisystemen bzw. Datenträgern, die nicht überprüft werden können oder sollen (Windows-Partitionen, CD-ROMs, DVDs, Disketten, virtuelle Dateisysteme, swap etc.), müssen Sie die Ziffer 0 angeben.

Wenn Einträge in der fünften und sechsten Spalte in `/etc/fstab` fehlen, wird 0 angenommen.

3.8.8 Spezielle Dateisysteme

Netzwerkdateisysteme

Das Einbinden von Unix- und Windows-Netzwerkverzeichnissen (NFS, Samba) wird in den Netzwerkkapiteln dieses Buchs beschrieben. Vorweg zwei typische `/etc/fstab`-Zeilen für ein NFS- und ein Samba-Verzeichnis:

```
# /etc/fstab: NFS, Samba
uranus:/data    /unixdata       nfs    user,exec         0  0
//venus/c       /windowsdata    smbfs  user,noauto,exec  0  0
```

Virtuelle Dateisysteme (proc, usbdevfs, devpts, shmfs)

Linux kennt eine Reihe virtueller Dateisysteme. Dabei handelt es sich um Linux-interne Verwaltungsinformationen, die wie Dateisysteme angesprochen werden. Tatsächlich existieren diese Dateien aber nicht auf irgendeiner Festplatte, sondern werden vom Kernel zur Verfügung gestellt. Linux bzw. die unter Linux laufenden Programme sind auf diese Dateisysteme angewiesen. Zurzeit sind vier derartige Dateisysteme häufig im Einsatz:

- Das `proc`-Dateisystem stellt Informationen über die Kernel- und Prozessverwaltung zur Verfügung (Verzeichnis `/proc`).
- Das `usbdevfs`- bzw. *usbfs*-Dateisystem hilft bei der Verwaltung von USB-Geräten (Verzeichnis `/proc/bus/usb`).
- Das `devpts`-Dateisystem stellt die Devices `/dev/pts` zur Verfügung. Dabei handelt es sich um virtuelle Pseudo-Terminals gemäß der Unix-98-Spezifikation, die von einigen Programmen zur Kommunikation benötigt werden.
- Das `shmfs` ist die Basis für *shared memory* gemäß System V. Es ermöglicht einen effizienten Datenaustausch zwischen zwei Programmen.

Es ist von der Distribution abhängig, ob diese Dateisysteme direkt im Rahmen des Init-V-Prozesses eingebunden werden oder ob dazu `fstab` ausgewertet wird. Aus diesem Grund fehlen in `fstab` je nach Distribution einige der folgenden Zeilen:

```
# /etc/fstab: virtuelle Dateisysteme zur internen Verwaltung
none      /dev/pts       devpts     mode=0620,gid=5   0 0
none      /proc          proc       defaults          0 0
```

```
none      /proc/bus/usb  usbdevfs   noauto          0 0
none      /dev/shm       tmpfs      defaults        0 0
```

Bei manchen Distributionen wird statt *none* der Name des Dateisystems angegeben. Diese Information wird in jedem Fall ignoriert, weil es für diese Dateisysteme keinen Device-Namen gibt. Informationen darüber, wie die virtuellen Verzeichnisse tatsächlich in den Verzeichnisbaum integriert sind, erhalten Sie mit `cat /etc/mtab`.

Swap-Partitionen und -Dateien

Das Einrichten einer Swap-Partition erfolgt normalerweise im Rahmen der Installation. Die dazugehörenden Zeilen in `/etc/fstab` sehen wie folgt aus:

```
# /etc/fstab: Swap-Partition
/dev/hdc7       swap           swap    pri=1 0 0
```

Ob und wie viel Swap-Speicher zur Verfügung steht bzw. tatsächlich verwendet wird, kann mit dem Kommando `free` überprüft werden. Wie viel Swap-Speicher ist sinnvoll? Die gängige Empfehlung lautet: etwa das Zweifache des RAMs. Wenn Sie also 512 MByte RAM besitzen, sollten Sie etwa 1 GByte Swap vorsehen.

Falls sich die Swap-Partition als zu klein herausstellt oder Sie aus anderen Gründen eine weitere Swap-Partition benötigen, richten Sie mit dem Programm `fdisk` eine neue Partition ein. Mit dem `fdisk`-Kommando `T` muss der Partitionstyp von 83 (Linux) auf 82 (Linux swap) gestellt werden. Nachdem die Partition mit `mkswap` formatiert wurde, kann sie mit `swapon` aktiviert werden. Wenn das klappt, sollte ein entsprechender Eintrag in `/etc/fstab` durchgeführt werden.

Das Einrichten einer Swap-Datei ist eine Notlösung, wenn für irgendein Programm mal die Swap-Partition zu klein ist. Der Zugriff auf eine Swap-Datei ist erheblich langsamer als der Zugriff auf eine eigene Swap-Partition.

Swap-Dateien werden üblicherweise im Verzeichnis `/dev` angelegt. Der erste Schritt besteht darin, mit dem Kommando `dd` eine leere Datei mit einer vorgegebenen Größe zu erzeugen. Die Größenangabe erfolgt in Blöcken, wobei die Blockgröße auf 1024 Bytes eingestellt wird. Anschließend wird die Swap-Datei wie eine Swap-Partition mit `mkswap` formatiert und mit `swapon` aktiviert. Das folgende Beispiel erzeugt eine recht kleine Swap-Datei von knapp einem MByte:

```
root#  dd bs=1024 if=/dev/zero of=/dev/swapfile count=1000
1000+0 records in
1000+0 records out
root#  mkswap /dev/swapfile 1000
Setting up swapspace, size = 1019904 bytes
root#  sync
root#  swapon -v /dev/swapfile
```

```
swapon on device /dev/swapfile
```

3.9 Dokumentation zu Linux

Zu Linux gibt es unglaublich viel Dokumentation, die teilweise gleich mitgeliefert wird und teilweise im Internet zu finden ist. Als letzte Rettung (für Experten) bietet sich schließlich der Blick in den Quellcode an, der oft ebenfalls gut dokumentiert ist.

Dieser Abschnitt gibt einen Überblick darüber, welche Dokumentationsdateien es gibt, wo sie sich befinden und wie sie gelesen werden können. Der Abschnitt geht auf die verschiedenen Formen der Online-Dokumentation ein, unter anderem auf `man`- und `info`-Texte, FAQs (Frequently Asked Questions), HOWTOs etc.

3.9.1 Hilfe in Gnome- und KDE-Programmen

Wenn Sie ein Problem bei einem Programm mit einer grafischen Benutzeroberfläche haben, sollten Sie als Erstes die jeweilige Online-Hilfe ausprobieren. Im Regelfall funktionieren (F1) bzw. das *Hilfe*-Menü wie in Windows-Programmen.

Unter KDE gelangen Sie außerdem über das KDE-Startmenü zum KDE-Hilfezentrum (`khelpcenter`). Dieses Programm gibt allgemeine Informationen zum KDE-Desktop-System. Es kann aber auch zum Lesen von `man`- und `info`-Texten verwendet werden (siehe unten) und bietet so einen zentralen Zugang zu sehr vielen Hilfetexten.

Auch unter Gnome führt das Startmenü zu einem Hilfezentrum: `gnome-help` bietet mit der Ausnahme der Volltextsuche ähnliche Funktionen wie `khelpcenter`.

3.9.2 Hilfe bei textorientierten Kommandos

Kommandos, die Sie üblicherweise in einem Konsolenfenster ausführen (z. B. `ls`, `cp` oder `top`), reagieren auf (F1) meist nicht und besitzen auch kein *Hilfe*-Menü. Es gibt aber natürlich auch für diese Kommandos Hilfetexte, die durch verschiedene Kommandos gelesen werden können:

- `kommando --help` liefert bei sehr vielen Kommandos eine Liste aller Optionen samt einer kurzen Erklärung zu ihrer Bedeutung.
- `man kommando` zeigt bei vielen Kommandos den `man`-Hilfetext an. Durch den meist mehrseitigen Text können Sie mit den Cursortasten blättern. (Q) beendet die Hilfe.

- `help kommando` funktioniert nur bei so genannten Shell-Kommandos, z. B. `cd` oder `alias`.
- `info kommando` ist eine Alternative zu `man`. Das `info`-System eignet sich vor allem für sehr umfangreiche Hilfetexte. Ob der Hilfetext im `man`- oder `info`-System vorliegt, hängt ganz einfach davon ab, für welches Hilfesystem sich die Programmentwickler entschieden haben. `man` ist aber deutlich populärer.

In den folgenden Abschnitten werden alle drei Varianten näher beschrieben.

3.9.3 man-Hilfetexte

`man` ist ein Programm zur Anzeige der Dokumentation vieler elementarer Kommandos wie `ls` oder `cp`. `man` wird meist in einer Textkonsole in der Form `man ls` aufgerufen (um den Hilfetext zu `ls` zu lesen).

```
man [optionen] [bereich] thema
```

Die optionale Angabe eines Bereichs schränkt die Suche nach `man`-Texten auf einen Themenbereich ein – etwa `man 3 printf`. Das ist dann notwendig, wenn mehrere gleichnamige `man`-Texte in unterschiedlichen Themenbereichen existieren. `man` zeigt in diesem Fall nur den ersten gefundenen `man`-Text an.

In vielen Unix- und Linux-Büchern werden zusammen mit den Kommandos gleich die `man`-Nummern angegeben – etwa `find(1)`. Damit wissen Sie sofort, wie Sie `man` aufrufen müssen. `man` kennt die Themenbereiche 1 bis 9 und n. (Manchmal werden Kommandos von Programmiersprachen in zusätzlichen Bereichen mit anderen Buchstaben eingeordnet.)

1	Benutzerkommandos
2	Systemaufrufe
3	Funktionen der Programmiersprache C
4	Dateiformate, Device-Dateien
5	Konfigurationsdateien
6	Spiele
7	Diverses
8	Kommandos zur Systemadministration
9	Kernel-Funktionen
n	Neue Kommandos

Außer dem Bereich können beim Aufruf von `man` mehrere Optionen angegeben werden:

`-a` zeigt der Reihe nach alle gleichnamigen `man`-Seiten an. (Ohne diese Option wird gewöhnlich nur die erste von mehreren gleichnamigen Dateien aus unterschiedlichen Themengebieten angezeigt. Bei manchen Linux-Distributionen gilt

-a aber als Default-Einstellung.)

-f *schlüsselwort* zeigt die Bedeutung eines Schlüsselworts an (einen einzeiligen Text). Mit dieser Option entspricht `man` dem Kommando `whatis thema`.

-k *schlüsselwort* zeigt eine Liste aller vorhandenen `man`-Texte an, in denen das Schlüsselwort vorkommt. Dabei wird allerdings keine Volltextsuche durchgeführt. Vielmehr werden nur die Schlüsselwörter jedes `man`-Textes analysiert. Mit dieser Option entspricht `man` dem Kommando `apropos thema`.

Wenn Sie alle gleichnamigen `man`-Texte (aus allen Bereichen) lesen möchten, müssen Sie `man` mit der Option `-a` verwenden. `man` zeigt jetzt den ersten gefundenen `man`-Text an. Sobald Sie den Text gelesen haben und `man` mit Q beenden, erscheint der `man`-Text zum nächsten Abschnitt.

Sobald ein `man`-Text einmal auf dem Bildschirm angezeigt wird, stehen – sofern als Anzeigeprogramm `less` eingestellt ist – unter anderem die folgenden Tastenkürzel zur Navigation im Text zur Verfügung:

man-Tastenkürzel	
Cursortasten	Text nach oben oder unten verschieben
Pos1, Ende	an den Beginn/das Ende des Textes springen
Y, E	Zeile nach oben oder nach unten
U, D	Seite nach oben (up) oder nach unten (down)
G, Shift+G	An den Beginn/das Ende des Textes springen
/ muster ↵	Vorwärts suchen
? muster ↵	Rückwärts suchen
N	Suche vorwärts wiederholen (next)
Shift+N	Suche rückwärts wiederholen
Q	Beenden (quit)
H	Hilfetext mit weiteren Tastenkürzeln anzeigen

man-Interna: Aus welchen Verzeichnissen `man` die Hilfetexte liest, kann wahlweise durch die Steuerungsdatei `/etc/manpath.config` oder durch die Umgebungsvariable *MANPATH* eingestellt werden.

Tipp

Unter KDE und Gnome können Sie `man`-Texte auch mit den jeweiligen Help- oder Webbrowsern lesen. Die folgenden Beispiele zeigen, wie Sie die `man`-Seite zu `ls` und ein Inhaltsverzeichnis aller `man`-Seiten anzeigen können.

```
user$ gnome-help man:ls
user$ khelpcenter man:ls
user$ khelpcenter 'man:(index)'
```

Im KDE-Programm `konqueror` können Sie `man`-Texte noch effizienter durch `#kommando` aufrufen.

Zu manchen Kommandos erhalten Sie Hilfe nicht mit `man`, sondern mit `help`. Das betrifft alle Kommandos, die direkt von der Shell ausgeführt werden. Beispielsweise liefert `help alias` eine kurze Beschreibung dieses Kommandos. `man alias` zeigt dagegen einen sehr viel längeren Hilfetext für sämtliche `bash`-Kommandos an.

3.9.4 info-Hilfetexte

`man`-Hilfetexte haben den Nachteil, dass sie nur schwer strukturierbar sind. Das alternative `info`-Format bietet hier deutlich bessere Möglichkeiten, weswegen vor allem umfangreiche Hilfetexte häufig nur in diesem Format vorliegen.

info `[optionen] [thema]`

`info` wird üblicherweise mit dem Parameter *thema* aufgerufen. Das Thema gibt an, welche Info-Datei aus dem `/usr/info`-Verzeichnis betrachtet werden soll. Wenn `info` ganz ohne Parameter gestartet wird, zeigt das Programm eine Übersicht der verfügbaren Hilfethemen an (Datei `/usr/info/dir`). Themen, die in `dir` nicht enthalten sind, können nur über die Option `-f` betrachtet werden! Das ist beispielsweise für die `elisp`-Dateien (Dokumentation zur Lisp-Programmierung für den Emacs) der Fall. `info` kann daher nicht automatisch alle `info`-Dateien bearbeiten, die sich in `/usr/info` befinden.

-f *datei* lädt die angegebene Datei statt einer Datei aus `/usr/info`. Im Gegensatz zum Thema muss der Dateiname vollständig angegeben werden (`jarg300.info.gz` und nicht nur `jarg300`). Wenn der `info`-Text auf mehrere Dateien verteilt ist, muss mit `-f` die erste Datei (etwa `elisp-1.gz`) angegeben werden.

Unter KDE bzw. Gnome können Sie `info`-Texte auch mit dem jeweiligen Hilfesystem lesen. Die folgenden Beispiele zeigen, wie der `info`-Text zu `emacs` sowie ein Inhaltsverzeichnis aller `info`-Texte angezeigt wird:

```
user$  gnome-help info:emacs
user$  gnome-help toc:info
user$  khelpcenter 'info:/emacs'
user$  khelpcenter 'info:/dir'
```

info-Tastenkürzel

Leertaste	Text nach unten scrollen
Backspace	Text nach oben scrollen
B, E	Zum Anfang/Ende der Info-Einheit springen (beginning/end)
Tab	Cursor zum nächsten Querverweis (*) bewegen
↩	Querverweis zu anderer Info-Einheit verfolgen
N	Nächste Info-Einheit derselben Hierarchiestufe (next)
P	Vorige Info-Einheit derselben Hierarchiestufe (previous)
U	Eine Hierarchieebene nach oben (up)
L	Zurück zum zuletzt angezeigten Text (last)
H	Ausführliche Bedienungsanleitung (help)
?	Kommandoübersicht
Q	Entfernt das Hilfefenster (quit)

3.9.5 Zusatzdokumentation zu Programmpaketen

Alle Programme Ihrer Distribution sind Teile von Paketen. Ein einzelnes Paket enthält meist nicht nur ein einziges Programm, sondern eine ganze Gruppe zusammengehöriger Programme, möglicherweise auch diverse Bibliotheken und Zusatzdateien. Viele Pakete enthalten außer `man`- oder `info`-Seiten eine Zusatzdokumentation, deren Installationsort distributionsspezifisch ist:

Debian, Red Hat, Fedora: `/usr/share/doc/paketname`
Suse: `/usr/share/doc/packages/paketname`

Zusatzdokumentation suchen: Was tun Sie, wenn Sie eine Zusatzdokumentation zu einem bestimmten Kommando suchen, aber nicht wissen, zu welchem Paket das Kommando gehört? Der erste Schritt besteht darin, den genauen Dateinamen des Kommandos festzustellen. Dazu führen Sie `which -a kommando` aus:

```
user$  which -a vi
/usr/bin/vi
```

Mit dieser Information können Sie ermitteln, zu welchem Paket `/bin/cp` gehört. Das folgende Kommando setzt voraus, dass Ihre Distribution RPM zur Paketverwaltung verwendet.

```
user$  rpm -qf /bin/cp
coreutils-5.0-90
```

Jetzt wissen Sie, dass `cp` ein Teil des Pakets `coreutils` ist. Ein weiterer Aufruf von `rpm` liefert eine Liste aller Dateien, die zur Dokumentation dieses Pakets existieren. Da das oft sehr viele Dateien sind, empfiehlt es sich, das Kommando `less` nachzustellen. Damit können Sie die Dateiliste in Ruhe lesen:

```
user$ rpm -qd coreutils — less
/usr/share/doc/packages/coreutils/NEWS
/usr/share/doc/packages/coreutils/README
/usr/share/info/coreutils.info.gz
/usr/share/man/man1/basename.1.gz
...
```

Bei Debian läuft es ähnlich ab, nur wird hier anstelle von `rpm` das Kommando `dpkg` verwendet. Viel häufiger verwendet man aber das Kommando `apt-cache`, mit dem man Installationspakete suchen kann. Dabei genügt die Angabe eines Teilstrings. Suchen wir mal den vi-Clone „elvis“:

```
user$ apt-cache search elvis
elvis - powerful clone of the vi/ex text editor (with X11 support)
elvis-common - common files for elvis, elvis-console and elvis-tools
elvis-console - powerful clone of the vi/ex text editor (without X11 support)
elvis-tiny - Tiny vi compatible editor for the base system.
elvis-tools - text editing tools for programmers (elvfmt, elvtags, ref)
global - Source code search and browse.
libwvstreams4.0-extras - C++ network libraries for rapid application development
nvi - 4.4BSD re-implementation of vi
```

Zusatzdokumentation lesen: Zusatzdokumentation liegt in den unterschiedlichsten Formaten vor. Die folgende Liste gibt einige Tipps, wie die Dokumente am besten gelesen werden können.

- Textformat: Einfache Textdateien können direkt mit `less` oder mit jedem beliebigen Editor gelesen werden.
- PostScript-Format (Kennung `*.ps`): Diese Dateien können mit jedem PostScript-Viewer gelesen werden: `ghostview`, `kghostview`, `gv`, `ggv` etc.
- Acrobat-Format (`*.pdf`): Hierbei handelt es sich um eine komprimierte Variante des PostScript-Formats. Zum Lesen können Sie alle PostScript-Viewer sowie `kpdf`, `xpdf` und `acroread` verwenden. (`acroread` ist per Default aber oft nicht installiert.)
- DVI-Format (`*.dvi`): Solche Dateien sind das Ergebnis einer TeX- oder LaTeX-Übersetzung und können mit `xdvi` oder `kdvi` gelesen werden. Über `dvips` ist eine Umwandlung in eine PostScript-Datei möglich.
- HTML-Format (`*.html`): Die Dateien können mit jedem WWW-Browser (Mozilla, Konquerer etc.) gelesen werden. In einer Textkonsole setzen Sie stattdessen `lynx` ein. Dieses Programm funktioniert auch im Textmodus (kann aber keine Bilder anzeigen).

Bei allen Varianten kann es vorkommen, dass die Datei komprimiert ist. Solche Dateien sind an der Dateierweiterung `.gz` oder `.bz2` zu erkennen. Zur Dekompression führen Sie `gunzip datei.gz` bzw. `bunzip2 datei.bz2` aus. Dadurch wird die komprimierte Datei durch eine entkomprimierte Version ersetzt.

3.9.6 Das Linux Documentation Project (LDP)

Das *Linux Documentation Project* hat sich das Ziel gesetzt, eine möglichst umfassende und zentrale Sammlung frei verfügbarer Linux-Dokumentationen zu schaffen. Tatsächlich ist die Informationsfülle auf der folgenden Seite eindrucksvoll:

http://www.tldp.org/

Die Mehrzahl der Dokumente ist in einem von drei Formaten erschienen: als HOWTO-Text (grundlagenorientierte Anleitungen), als FAQ-Text (Fragen und Antworten) oder als Guide (Buchform).

HOWTOs: HOWTO-Texte sind systematisch aufgebaute Grundlagentexte zu bestimmten Themen. HOWTO-Texte zählen zu den wichtigsten Informationsquellen, besonders was die Installation oder Konfiguration spezieller Hard- oder Software betrifft. Allerdings sind die Texte zum Teil sehr umfangreich und enhalten weit mehr Informationen, als Sie eigentlich haben wollten. Beachten Sie auch, dass manche HOWTOs veraltet sind und nicht mehr gewartet werden.

Bei manchen Distributionen werden die HOWTO-Texte in einem Paket mitgeliefert. Dieses Paket muss aber oft eigens installiert werden. Ansonsten finden Sie die HOWTO-Texte im Internet:

http://www.tldp.org/docs.html#howto
http://www.linuxhaven.de/dlhp/ (deutsche Übersetzungen)

FAQs: Im Vergleich zu den HOWTOs spielen FAQ-Dokumente – also *Frequently Asked Questions* samt Antworten – eine untergeordnete Rolle. Die folgenden Links verweisen auf die wichtigsten Linux-FAQs in deutscher und englischer Sprache:

http://www.tldp.org/FAQ/
http://www.linuxfaq.de
http://tux.bartolich.at/at.linux-FAQ/html/toc.html

LDP-Guides: Im Rahmen des *Linux Documentation Project* sind ganze Bücher entstanden, die vereinzelt auch in gedruckter Form veröffentlicht wurden:

http://www.tldp.org/guides.html

3.9.7 Kernel-Dokumentation

Eine Menge hardware-spezifischer Informationen finden Sie im Verzeichnis `/usr/src/linux/Documentation` (nur wenn Sie den Kernel-Code installiert haben). Die Informationen setzen zumeist ein recht gutes Grundwissen zur Hardware Ihres Rechners und zum Teil auch zur Programmierung voraus. Die Mühe beim Lesen lohnt sich aber oft! Selten finden Sie derart fundierte Informationen über die Interna Ihres Rechners. (Übrigens ist auch der Kernel-Code selbst recht gut dokumentiert; wenn Sie ein bestimmtes Hardware-Problem haben, kann ein Blick in die entsprechenden `*.c`-Dateien nicht schaden.)

Tipp

Mit etwas Glück brauchen Sie nicht gleich den gesamten Kernel-Code zu installieren, der weit über 100 MByte Ihrer Festplatte füllt. Stattdessen bieten manche Distributionen eigene Kernel-Dokumentations-Packages an.

3.9.8 Weitere Dokumentationen im Internet

Linux-Websites: Neben den bereits erwähnten Informationsquellen gibt es natürlich unzählige Websites, auf denen Sie weitere Informationen zu Linux finden. Die folgenden Links verweisen auf einige besonders hilfreiche Seiten:

http://www.selflinux.de/
http://www.linuxfibel.de/
http://www.dsl.org/cookbook/

Newsgroups: Es gibt zahllose Newsgroups zum Thema Linux. Auch wenn Sie die darin stattfindenden Diskussionen nicht ständig verfolgen, können Sie sehr bequem in einem riesigen Archiv alter News-Beiträge suchen. Wenn Sie irgendwelche Konfigurations- oder Hardware-Probleme haben, stoßen Sie hier mit sehr großer Wahrscheinlichkeit auf andere Linux-Anwender, die dieselben Probleme auch schon hatten (und mit etwas Glück auch lösen konnten). Die populärste Suchmaschine für Newsgroups ist Google:

http://groups.google.com

RFCs: RFC steht für *request for comment*. Dahinter verbergen sich Dokumente, die diverse Protokolle (darunter z. B. TCP, IP, FTP, PPP etc.) im Detail beschreiben. Das etwas merkwürdige Kürzel RFC deutet auf die Entstehungsgeschichte dieser Protokolle hin: Sie wurden im Regelfall nicht durch eine Person, Organisation oder Firma diktiert, sondern sind aus einem (oft langwierigen) Diskussionsprozess entstanden. Die hier dargestellten Informationen sind eher technischer Natur:

http://www.faqs.org/rfcs/

Kapitel 4

Die Linux-Shell

Einführung

In diesem Kapitel geht es um ein Programm, das bei der Anwendung einer grafischen Benutzerschnittstelle anfangs oft unbemerkt bleibt: der Kommandointerpreter, „Shell“ genannt, der ein mächtiges Werkzeug zur Automatisierung regelmäßig wiederkehrender Aufgaben ist. Wir haben die Shell auch schon ganz kurz im Abschnitt „Textkonsolen und Shell-Fenster“ erwähnt. Was dort gesagt wurde, gilt natürlich besonders in diesem Kapitel – insbesondere auch die Tastenkürzel.
Die Shell bietet neben ihren Diensten als interaktiver Kommandointerpreter eine mächtige Sprache zum Erstellen so genannter Shell-Skripten (*shell scripts*). So wie bei der Programmiersprache *C* der eigentliche Sprachumfang recht gering und die Mächtigkeit der Sprache durch die zahllosen Bibliotheksfunktionen gegeben ist, „lebt“ die Shell v. a. von den vielen, vielen Linux-Kommandos, die alle kommandozeilenorientiert aufgerufen werden können. Dies ist auch das Ziel dieses Kapitels. Worauf deshalb nicht besonders eingegangen wird, sind die interaktiven Möglichkeiten der Shell (wie z. B. History-Funktion, Aliase, Zeileneditor etc.). Auch beschränkt sich das Buch auf die „Standard-Shell“ von Linux, die *bash*. Die erste Linux-Shell, die nach ihrem Entwickler, Steven R. Bourne, benannte *Bourne-Shell* diente als Vorbild, weshalb „bash“ auch ein Acronym für „Bourne again shell“ ist (wenn Sie das laut aussprechen, wird auch das verborgene Wortspiel deutlich).

Der Vorteil bei der Shell-Programmierung ist auch die Interaktivität der Shell. Sie können alle (!) Sprachkonstruktionen auch auf der Kommandozeile ausführen lassen und natürlich können Sie durch abwechselndes Probieren und Erweitern Ihr Shell-Programm Schritt für Schritt entwickeln – fast so wie früher mit *BASIC* auf dem C64.

4.1 Aufgaben der Shell

Die Hauptaufgaben bzw. Anwendungsmöglichkeiten der Shell lassen sich in folgenden Punkten zusammenfassen:

- Kommandozeile vom Terminal annehmen, das Kommando entschlüsseln und in seine Komponenten (Kommandoname, Optionen, Parameter) aufteilen
- Gewünschtes Programm suchen und starten; der Suchweg wird durch eine Shell-Variable `PATH` vorgegeben.

  Mittels `which Programmname` kann man den Pfad zu einem Programm feststellen bzw. untersuchen, ob sich das Programm im Pfad befindet. Das Kommando `type` arbeitet ähnlich, es findet zusätzlich auch alle internen Kommandos der Shell.
- Steuerung der Ein- und Ausgabe des Terminals; UNIX (und so auch Linux) ist als Dialogsystem konzipiert und war ursprünglich für den Betrieb mit Fernschreibern (*Teletype*) und als Datensichtgerät konzipiert. Diese so genannten Terminals findet man noch in den *logischen* Terminals, z. B.:
  - den Konsolen, die dem Bildschirm und der Tastatur des PCs zugeordnet sind,
  - den Pseudo-Terminals, die einer Telnet-Verbindung zugeordnet werden oder
  - den X-Terminal-Fenstern auf der grafischen Benutzeroberfläche.

  Linux behandelt alle Ein- und Ausgabemedien als Dateien, auch Geräte wie Terminals, Drucker, serielle Schnittstellen usw. Daher gibt es drei Standarddateien, die dem aktuellen Terminal zugeordnet sind:
  - Standardeingabe (Tastatur, `stdin`)
  - Standardausgabe (Bildschirm, `stdout`)
  - Standardfehlerausgabe (Bildschirm, `stderr`)

  Die Shell kann angewiesen werden, die Ein- und Ausgaben auf Geräte oder auf Dateien umzuleiten. Man kann also beispielsweise Shell-Eingaben auch aus einer Datei lesen oder die Ausgabe der gestarteten Programme in einer Datei speichern.
- Ersetzung von Sonderzeichen und Befehlssubstitution; Kommandoeingaben können durch Sonderzeichen erweitert und der Kommandostring kann selbst durch die Shell expandiert werden.
- Ablaufsteuerung; die Shell beherrscht viele Strukturen, wie man sie von Programmiersprachen her kennt (Test, bedingte Anweisung, Schleifen, Variablen, Rechenanweisungen).

- Erzeugen so genannter Kind- und Hintergrundprozesse. Es besteht die Möglichkeit, Programme „im Hintergrund“ zu starten und am Bildschirm weiterzuarbeiten, während das Programm läuft.

4.2 Grundlegende Eigenschaften der Shell

Folgende Punkte mögen – in zugegeben knapper Form – zusammenfassen, mit welcher Art von Werkzeug man es bei der Shell zu tun hat und nach welchen Prinzipien sie arbeitet:

1. Sobald die Shell bereit ist, Kommando-Eingaben anzunehmen, meldet sie sich mit einem Bereitschaftszeichen (*Prompt*). Im einfachsten Fall ist dies – wie Sie im dritten Kapitel gesehen haben – ein `$`-Zeichen für den Normaluser und das `#` für den Superuser. Das Zeichen `>` erscheint, wenn noch weitere Eingaben erwartet werden. Wie vieles in der Shell kann der Prompt beliebig modifiziert werden.
2. Die Shell ist ein ganz „normales“ Programm, das von der Standardeingabe (Tastatur) liest und auf die Standardausgabe (Bildschirm) ausgibt. Wenn das Dateiende erreicht wird (*End of File*, `EOF`), z. B. durch Eingabe von (Strg)-(D), terminiert sie. Handelt es sich um die Login-Shell, erfolgt ein Logoff des Benutzers.
3. Die Shell kann wie ein Programm als Subshell aufgerufen werden (Schachtelung). Dies wird beispielsweise benötigt, um Shell-Programme (*shell scripts*) zu testen.
4. Es gibt zwei Möglichkeiten, den Pfad für ein Kommando festzulegen:
   - direkt als absoluter oder relativer Pfadname,
   - indirekt über den Pfad, der in der Shell-Variablen `PATH` gespeichert ist.

Zur Erinnerung: In der Literatur (auch beim Kommando `man`) werden die Kommandos in Kurzform beschrieben. Dabei werden Werte, die optional sind, in eckige Klammern gesetzt. Die (meist aus einem Buchstaben bestehenden) Optionen werden als Kette aufgelistet, z. B. `[-aeGhlz]`. Das bedeutet nichts anderes, als dass eine beliebige Kombination dieser Optionen möglich ist (ob sie sinnvoll ist, wird dabei nicht berücksichtigt).

Ein Kommando wird erst ausgeführt, wenn der Benutzer am Ende der Kommandozeile die (↵)-Taste drückt. Eine genauere Kenntnis des dann folgenden Ablaufs erlaubt es zu verstehen, warum etwas nicht zu dem Ergebnis führt, das man erwartet hat. Daher sollen hier die einzelnen Schritte dieses Arbeitsablaufs kurz beschrieben werden (einige der Kommando-Trenner, Jokerzeichen und Variablen werden später behandelt):

1. Die Shell liest die Eingabe bis zum ersten Kommando-Trenner und stellt fest, ob Variablenzuweisungen erfolgen oder die Ein-/Ausgabe umgelenkt werden soll.
2. Die Shell zerlegt die Kommandozeile in Argumente. Sie trennt die einzelnen Argumente durch eines der Zeichen, die in der Shell-Variablen `IFS` (*Internal Field Separator*) stehen, normalerweise Leerzeichen, Tabs und Newline-Zeichen.
3. Variablenreferenzen, die in der Kommandozeile stehen, werden durch ihre Werte ersetzt.
4. Kommandos, die in `` `...` `` oder bei der Bash in `$(...)` stehen, werden ausgeführt und durch ihre Ausgabe ersetzt.
5. `stdin`, `stdout` und `stderr` werden auf ihre „Zieldateien" umgelenkt (s. unten).
6. Falls in der Kommandozeile noch Zuweisungen an Variablen stehen, werden diese ausgeführt.
7. Die Shell sucht nach Jokerzeichen (Metazeichen) und ersetzt diese durch passende Dateinamen.
8. Die Shell führt das Kommando aus.

4.3 Ein-und Ausgabeumleitung

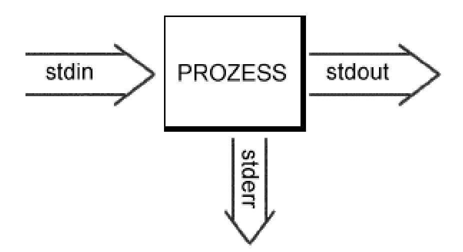


Abbildung 4.1: Standarddateien unter Linux

Die drei dem Terminal zugeordneten Dateikanäle (Pseudodateien) `stdin`, `stdout` und `stderr` können jederzeit auf Dateien oder Geräte umgeleitet werden. Für C-Programmierer: Den drei Standarddateien sind die Filehandles `0` (`stdin`), `1` (`stdout`) und `2` (`stderr`) zugeordnet.

4.3.1 Eingabeumleitung

Das Programm liest nun nicht mehr von der Tastatur (`stdin`), sondern aus einer Datei, bis das Dateiende erreicht ist.

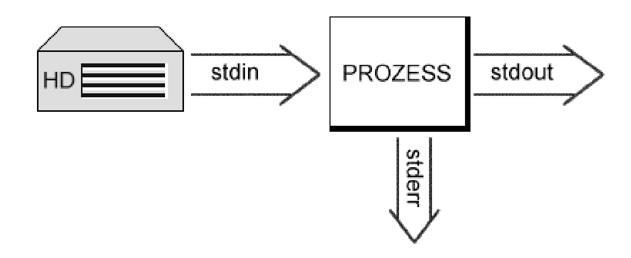


Abbildung 4.2: Umleitung der Standardeingabe

Die Eingabeumleitung erfolgt durch das Zeichen „`<`“, gefolgt von einem Dateinamen:

```
Kommando < Dateiname
```

Statt z. B. beim `mailx`-Kommando den Text direkt einzugeben, kann eine Datei an den Empfänger gesendet werden:

`user$` **mailx -s "Automatische Nachricht" linus < Nachricht**

4.3.2 Ausgabeumleitung

Die Ausgabe des Programms wird nicht auf dem Bildschirm (`stdout`) ausgegeben, sondern in eine Datei geschrieben. Die Ausgabeumleitung erfolgt durch das Zeichen „`>`“, gefolgt von einem Dateinamen.

Falls die Datei noch nicht vorhanden war, wird sie automatisch angelegt. Falls die Datei schon vorhanden ist, wird sie überschrieben, d. h. es wird immer ab dem Dateianfang geschrieben:

```
Kommando > Dateiname
```

Beispiel: Ausgabe der Verzeichnisbelegung in eine Datei namens `info`:

`user$` **ls -l > info**

Fehlermeldungen erscheinen nach wie vor auf dem Bildschirm. Die Umleitung der Fehlerausgabe `stderr` erfolgt ebenso wie die Ausgabeumleitung, jedoch wird hier die Zeichenfolge `2>` verwendet, da `stderr` die Handle-Nummer `2` hat.

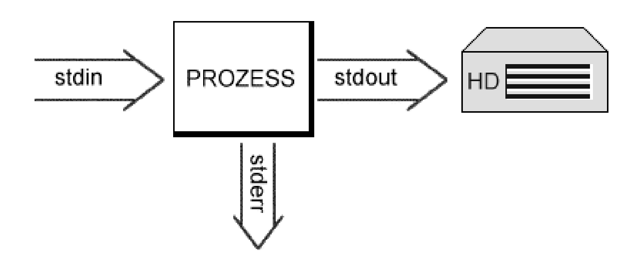


Abbildung 4.3: Umleitung der Standardausgabe

```
Kommando 2> Fehlerdatei
```

(Die Umleitung der Standardausgabe ist nur die Kurzform von *Kommando* `1>` *Dateiname*). Natürlich ist eine beliebige Kombination von Ein- und Ausgabeumleitung möglich, z. B.

```
Kommando < Eingabedatei > Ausgabedatei 2> Fehlerdatei
```

Es ist darüber hinaus auch möglich, die Ausgabe des Programms an eine bereits vorhandene Datei anzuhängen. Dazu wird das „`>`“ doppelt geschrieben.

```
Kommando >> Sammeldatei
```

Dazu einige Beispiele:

Dateiliste und aktive Benutzer in eine Datei schreiben:

```
user$  ls -l > liste
user$  who >> liste
```

Die Umleitung von Ein- und Ausgabe lässt sich auch unterdrücken. Für die Ausgabe schreibt man dann

```
Kommando > /dev/null
```

oder für die Fehlerausgabe

```
Kommando 2> /dev/null
```

Beides lässt sich auch kombinieren:

```
Kommando > Ergebnisdatei 2> /dev/null
```

Will man ein Programm mit einem beliebigen Eingabedatenstrom versorgen, schreibt man

```
Kommando < /dev/zero
```

Die Umleitung von `stdout` und `stderr` in dieselbe Datei würde prinzipiell eine zweimalige Angabe der Datei (eventuell mit einem langen Pfad) erfordern. Für die Standarddateien werden in solchen Fällen spezielle Platzhalter verwendet:

```
Kommando > Ausgabe 2>&1
```

Es gibt drei Platzhalter für die Standarddateien: `&0` für die Standard*eingabe*, `&1` für die Standard*ausgabe* und `&2` für die Standard*fehlerausgabe*. Es ist sogar möglich, in Shell-Skripten anonyme Dateien zu öffnen, die nur über die Dateinummer (`&4, &5, ...`) referenziert werden.

Hinweis

Bei der Ein- und Ausgabeumleitung darf nicht die gleiche Datei für Eingabe und Ausgabe verwendet werden – sie wird sonst verwüstet. So liefert beispielsweise `sort < dat > dat` keineswegs eine sortierte Datei `dat`, sondern die Datei wird gelöscht.

4.3.3 Pipes

Eine Pipe verbindet zwei Kommandos über einen temporären Puffer, d. h. die Ausgabe vom ersten Programm wird als Eingabe vom zweiten Programm verwendet. Alles, was das erste Programm in den Puffer schreibt, wird in der gleichen Reihenfolge vom zweiten Programm gelesen. Pufferung und Synchronisation werden vom Betriebssystem vorgenommen. Der Ablauf beider Prozesse kann verschränkt erfolgen. In einer Kommandofolge können mehrere Pipes vorkommen. Der Pipe-Mechanismus wird durch das Zeichen | (senkrechter Strich) aktiviert:

```
Kommando1 | Kommando2
```

Beispiel: Ausgabe der Dateien eines Verzeichnisses mit der Möglichkeit zu blättern:

Weil die Zeile *Kommando* `<` *Eingabedatei* `>` *Ausgabedatei* `2>` *Fehlerdatei* optisch etwas seltsam wirkt, bemühen viele Shell-Programmierer – insbesondere in Verbindung mit den Pipes – das `cat`-Kommando. „cat“ steht für „concatenate“ (= zufügen, aneinderhängen) und genau das macht das Kommando. Es gibt die Inhalte der Datei(en), die als Parameter angegeben wurden, auf der Standardausgabe aus. Aus dem Kommando oben wird dann:

```
cat Eingabedatei | Kommando > Ausgabedatei 2> Fehlerdatei
```

Natürlich können auch mehrere Kommandos hintereinander durch Pipes verbunden werden:

```
Kommando1 | Kommando2 | Kommando3 | Kommando4 | ...
```

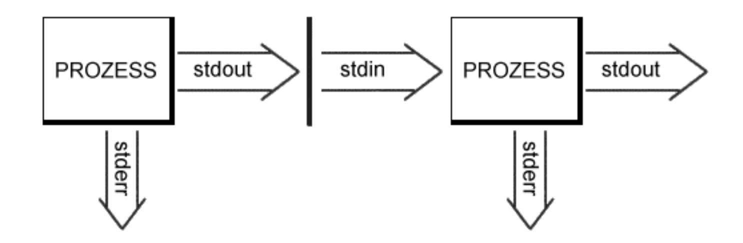


Abbildung 4.4: Pipe: Die Ausgabe eines Programms wird in die Eingabe des nächsten geleitet.

Solche Kommandofolgen werden auch als *Filter* bezeichnet. Einige nützliche Filter sind in jedem Linux-System verfügbar. Zum Beispiel:

- `head [-n]` [*datei(en)*]

  Ausgabe der ersten `n` Zeilen aus den angegebenen Dateien. Voreinstellung ist zehn Zeilen. Wird keine Datei angegeben, liest `head` von der Standardeingabe.

- `tail [-/+n] [bc[f|r]]` [*datei*]

  Ausgabe der letzten `n` Zeilen einer Datei. Voreinstellung für `n` ist 10. Wird keine Datei angegeben, liest `tail` von der Standardeingabe. Optionen:

  `+n` ab der `n`-ten Zeile ausgeben

  `-n` die letzten `n` Zeilen ausgeben. Wird hinter die Zahl `n` ein `b` gesetzt (z. B. `-15b`), werden nicht `n` Zeilen, sondern `n` Blöcke ausgegeben. Wird hinter die Zahl `n` ein `c` gesetzt (z. B. `-200c`), werden nicht `n` Zeilen, sondern `n` Zeichen (*characters*) ausgegeben.

  `-r` Zeilen in umgekehrter Reihenfolge ausgeben (letzte zuerst). Funktioniert nicht bei GNU-`tail` – stattdessen kann man das Programm `toc` verwenden.

  `-f` `tail` am Dateiende nicht beenden, sondern auf weitere Zeilen warten (Ende des Kommandos mit Strg-C). Damit kann man z. B. Logfiles beobachten, die ständig wachsen.

- `tee [-i] [-a]` [*datei*]

  Pipe mit T-Stück: Kopiert von `stdin` nach `stdout` und schreibt die Daten gleichzeitig in die angegebene Datei. Optionen:

  `-i` Ignorieren von Interrupts (Unterbrechungs-Taste)

  `-a` Anhängen der Info an die angegebene Datei (Voreinstellung: Überschreiben der Datei)

- `wc [-lwc] [`*Datei(en)*`]`

  Dieses Kommando zählt Zeilen, Worte oder Zeichen in einer Datei. Wird kein Dateiname angegeben, liest `wc` von der Standardeingabe. Normalerweise zählt man damit in Skripten irgendwelche Ergebnisse. Optionen:

  `-l` Zähle Zeilen

  `-w` Zähle Worte

  `-c` Zähle Zeichen

Weitere Filter sind `more, less, tr, ...`

4.4 Metazeichen zur Expansion von Dateinamen

Damit bei der Angabe von z. B. Dateinamen nicht jeder einzelne Name getippt werden muss, sondern die Dateien alle oder nach bestimmten Kriterien ausgewählt werden können, gibt es so genannte *Metazeichen* (auch *Jokerzeichen* oder *Wildcards* genannt).

Metazeichen	Bedeutung
`*`	Der Stern steht für eine beliebige Zeichenfolge – oder für überhaupt kein Zeichen. Dazu ein Beispiel: `ab*` steht für alle Dateinamen, die mit `ab` beginnen, auch für `ab` selbst (also `ab`, `abc`, `abcd`, `abxyz` usw.).
`?`	Das Fragezeichen steht für genau ein beliebiges Zeichen. Zum Beispiel: `?bc` steht für alle Dateinamen mit 3 Zeichen, die auf `bc` enden (also `abc`, `bbc`, `1bc`, `vbc`, `xbc` usw.), nicht jedoch für `bc`.
`[ ]`	Die eckige Klammer wird ersetzt durch *eines* der in der Klammer stehenden Zeichen. Auch ein Bereich ist möglich, z. B. `[a-k]` = `[abcdefghijk]`. Beispiel: `a[bcd]` wird ersetzt durch `ab`, `ac` und `ad`.
`[! ]`	Die eckige Klammer mit Ausrufezeichen wird ersetzt durch ein in der Klammer *nicht* stehendes Zeichen, zum Beispiel: `[!abc]` wird ersetzt durch ein beliebiges Zeichen außer `a`, `b` oder `c`.
\	Der *backslash* hebt den Ersetzungsmechanismus für das folgende Zeichen auf. Beispiel: `ab\?cd` wird zu `ab?cd` – das Fragezeichen wird übernommen.

Tabelle 4.1: Tabelle der Shell-Metazeichen

Im Gegensatz zu anderen Systemen (z. B. MS-DOS) werden diese von der Shell ersetzt. Dies ist eine sehr wichtige Tatsache, die zur Folge hat, dass nahezu jedes Linux-Kommando als Dateiangabe immer eine (im Rahmen der Betriebssystem-Parameter) beliebige Menge von Dateien als Parameter haben kann. Im Programm sind daher auch keine Systemaufrufe notwendig, die auf die Verzeichnisinformation zugreifen; es wird lediglich eine Schleife benötigt, welche die einzelnen Dateien nacheinander bearbeitet. Metazeichen sind Zeichen mit erweiterter Bedeutung. Die Shell ersetzt die Metazeichen durch alle Dateinamen des aktuellen Verzeichnisses, die auf das Muster passen. Dabei können die Metazeichen beliebig oft an beliebiger Stelle im Dateinamen stehen (z. B.: `*abc*def*`). Die Metazeichen sind in der Tabelle 4.1 aufgelistet.

Wichtig: Bei der Umleitung von Ein- und Ausgabe werden Metazeichen in den Dateinamen hinter dem Umleitungszeichen nicht ersetzt. Beispiele für die Anwendung:

`user$` **ls -l a***

listet alle Dateien, die mit `a` anfangen.

`user$` **ls test?**

listet alle Dateien, die mit `test` anfangen und fünf Zeichen lang sind (`test1`, `test2`, `testa` usw.).

`user$` **ls /dev/tty1[1-9]**

listet alle Terminalbezeichnungen mit einer `1` in der Zehnerstelle (`tty11`, `tty12`, . . . , `tty19`)

> **Hinweis**
>
> Der `*` ist ein gefährliches Zeichen, denn Tippfehler können zum Fiasko führen, wenn versehentlich ein Leerzeichen zuviel eingegeben wird.
>
> `user$` **rm a***
>
> löscht beispielsweise alle Dateien, die mit `a` beginnen.
>
> `user$` **rm a ***
>
> löscht dagegen erst die Datei `a` und dann alle Dateien im Verzeichnis.

Wie schon im dritten Kapitel erwähnt, gilt:

- Der Punkt am Anfang von Dateinamen stellt eine Ausnahme dar; er muss explizit angegeben werden, einerseits wegen der Verzeichnisreferenzen `.` bzw. `..`, andererseits weil Dateien, die mit einem Punkt beginnen, normalerweise nicht angezeigt werden.
- Der \ am Zeilenende unterdrückt auch das ⏎-Zeichen – das Kommando kann in der folgenden Zeile fortgesetzt werden (es erscheint dann der Prompt `>` anstelle von `$`).

4.5 String-Ersetzungen (*Quoting*)

Um bestimmte Sonderzeichen (z. B. `*`, `?`, `[ ]`, Leerzeichen, Punkt) zu übergeben, ohne dass sie von der Shell durch Dateinamen ersetzt werden, werden Anführungszeichen verwendet, die auch ineinander geschachtelt werden können. Dabei haben die drei verschiedenen Anführungszeichen *Doublequote* (¨), *Quote* (´) und *Backquote* (`) unterschiedliche Bedeutung:

- `¨...¨`
  Keine Ersetzung der Metazeichen `* ? [ ]`, jedoch Ersetzung von Shell-Variablen (siehe unten) und Ersetzung durch die Ergebnisse von Kommandos (*Backquote*). Auch \ funktioniert weiterhin. Dazu ein Beispiel:

  ```
  echo Der * wird durch alle Dateinamen ersetzt}
  echo "Der * wird hier nicht ersetzt"
  ```

- `´...´`
  Das einfache Anführungszeichen unterdrückt jede Substitution. Zum Beispiel:

  ```
  echo ´Weder * noch `pwd` werden ersetzt´
  ```

- `` `...` ``
  Zwischen *Backquotes* gesetzte Kommandos werden ausgeführt und das Ergebnis wird dann als Parameter übergeben (d. h. die Ausgabe des Kommandos landet als Parameter in der Kommandozeile). Dabei werden Zeilenwechsel zu Leerzeichen. Braucht dieses Kommando Parameter, tritt die normale Parameterersetzung in Kraft. Zum Beispiel:

  ```
  echo ¨Aktuelles Verzeichnis: ` pwd `¨
  ```

  Weil die verschiedenen `Quotes` manchmal schwer zu unterscheiden sind, wurde bei der `bash` eine weitere Möglichkeit eingeführt: Statt in *Backquotes* wird die Kommandofolge in `$(...)` eingeschlossen., z. B.:

  ```
  echo ¨Aktuelles Verzeichnis: $(pwd)¨
  ```

4.6 Bash-Spezialitäten

Was wir bisher beschrieben haben, ist bereits seit der ersten Bourne-Shell `/bin/sh` gültig. Die Bash ist vollständig kompatibel zu der originalen *Bourne-Shell*, aber sie hat etliche Erweiterungen erfahren.

Wie bereits erwähnt, kann man die Eingaben eines Programms auch aus einer Datei lesen, anstatt sie von Hand einzugeben. Die Eingaben der Shell sind in der Regel Kommandos. Die Shell liest unter gewissen Bedingungen auch automatisch Kommandos aus bestimmten Dateien und arbeitet sie ab. DOS-Benutzer kennen etwas Vergleichbares von der Datei *AUTOEXEC.BAT*.

Im Verlauf des ersten Aufrufs arbeitet jede Shell zunächst die Datei `/etc/profile` ab, in der der Systemverwalter für alle Anwender gültige Kommandos und Variablen eintragen kann. Daran anschließend sucht die Bash nach einer der folgenden Dateien im Heimatverzeichnis des Benutzers und führt diejenige aus, die zuerst gefunden wird:

1. `.bash_profile`
2. `.bash_login`
3. `.profile`

Anders verhält es sich jedoch, wenn die Bash nicht als Login-Shell gestartet wird. Dann führt sie ausschließlich die Datei `.bashrc` aus.

Die Ausführung der Dateien lässt sich über zwei Kommandozeilen-Parameter steuern. Mit dem Parameter `-noprofile` veranlassen Sie, dass die Bash keine der oben genannten Startdateien ausführt, mit `-norc` erreichen Sie, dass die persönliche Konfigurationsdatei `~/.bashrc` ignoriert wird.

4.6.1 Der Prompt

Die Bash gibt einen Eingabe-Prompt aus, häufig in der Form *Username*:*Pfad*`$` Der Prompt ist über die Umgebungsvariable `PS1` konfigurierbar. Ein Prompt in der oben genannten Form resultiert aus der Einstellung

```
PS1=\u:\w\\$
```

Um andere Einstellungen auszuprobieren, müssen Sie die Variable neu belegen. Innerhalb dieser Einstellungen können Sie folgende Metazeichen verwenden:

Metazeichen und ihre Bedeutung	
`\h`	Rechnername ohne Domäne
`\H`	Vollständiger Rechnername
`\t`	Aktuelle Uhrzeit
`\u`	User-Name
`\w`	Aktuelles Verzeichnis
`\#`	Fortlaufende Nummer des aktuellen Befehls
`\\`	das Zeichen \
`\$`	`#` für `root`, `$` für normale User

4.6.2 Editieren der Kommandozeile

Die Hauptaufgabe einer Shell ist die Entgegennahme und Ausführung von Kommandos. Zum Bearbeiten der aktuellen Kommandozeile stehen sowohl *Emacs*- als auch *vi*-kompatible Editiermodi zur Verfügung – voreingestellt ist der *Emacs*-Modus. Wenn Sie lieber im *vi*-Modus arbeiten, stellen Sie ihn durch den Befehl `set -o vi` ein, zurück in den *Emacs*-Modus geht's mit `set -o emacs`.

Die folgende Tabelle stellt die wichtigsten Editierfunktionen des *Emacs*-Modus zusammen:

Tasten und ihre Wirkung	
← und →	Bewegen des Cursors in der Kommandozeile
Esc F	springt zum nächsten Wortende
Esc B	springt zum vorigen Wortanfang
Strg+E	springt zum Zeilenende
Strg+A	springt zum Zeilenanfang
Strg+K	löscht den Text ab der Cursorposition bis zum Ende der Zeile
Strg+Y	fügt den zuletzt gelöschten Text nach der Cursorposition ein
Tab	ergänzt ein Eingabefragment zum passenden Dateinamen Ist die Ergänzung nicht eindeutig möglich, ertönt ein Signal und Sie erhalten durch zweimaliges Drücken der Tab-Taste eine Liste der in Frage kommenden Dateinamen.

Abhängig von der eingesetzten Terminalemulation können Sie Kombinationen mit der Esc-Taste oft auch mit der Alt-Taste nachbilden. Statt also nacheinander Esc und F zu drücken, funktioniert meist auch die Kombination Alt+F.

4.6.3 History-Mechanismus

In der Bash können Sie nicht nur die aktuelle Kommandozeile editieren, sie speichert auch alle zuvor eingegebenen Befehle in einer Datei, der so genannten Kommandozeilen-*History*. Auch diese Datei befindet sich im `home`-Directory des Benutzers und heißt `.bash_history`. Mit den Cursortasten ↑ und ↓ kann man in dieser Liste blättern. Darüber hinaus stehen folgende History-Befehle zur Verfügung: Strg+R sucht nach dem letzten Kommando anhand der Anfangsbuchstaben, ← springt an den Anfang der History-Liste und → springt ans Ende der History-Liste. Die Anzahl der Befehlszeilen wird mit der Variablen `HISTSIZE` eingestellt. Wächst die History darüber hinaus, verwirft die Bash die ältesten Zeilen.

4.6.4 Wichtige interne Kommandos

Interne Kommandos sind direkt in die `bash` eingebaut. Es muss also kein externes Programm geladen und gestartet werden und so ist die Kommandoausführung schneller und speicherschonender.

- `alias` weist einem Befehl oder einer Befehlsfolge einen neuen Namen zu. Beispiele für Windows-Umsteiger sind:

  ```
  alias dir='ls -l'
  alias cd..='cd ..'
  alias md='mkdir'
  alias rd='rmdir'
  alias del='rm -i'
  ```

  Ohne Parameter gibt dieser Befehl eine Liste der aktuell definierten Alias-Namen aus. Zum Löschen eines Alias-Eintrags verwendet man den Befehl `unalias` *Name*.

- `fg`, `bg` und `jobs` sind Befehle für die interne Jobkontrolle der Bash. Ein laufendes Programm kann mit (Strg)+(Z) in den Hintergrund verschoben werden; dort wird es zunächst angehalten (*suspend*). Das Kommando `jobs` gibt eine Liste aller derzeit im Hintergrund „schlafenden" Programme aus. Dabei wird für jedes Programm die interne Jobnummer angegeben. Es ist wichtig anzumerken, dass die vergebene Jobnummer in keinem Zusammenhang zu der – etwa von `ps` – angezeigten Prozessnummer steht. Mit `fg` *Jobnummer* holen Sie eine Task wieder in den Vordergrund. Soll ein angehaltenes Kommando dagegen im Hintergrund weiterlaufen, verwendet man den Befehl `bg` *Jobnummer*.

- `dirs`, `pushd`, `popd` verwalten den Stapel der internen Verzeichnisse. Mit `pushd` *Verzeichnis* wird das angegebene Verzeichnis auf dem Stapel abgelegt. Mit `popd` gelangen Sie anschließend zum zuletzt abgelegten Verzeichnis zurück. Den aktuellen Stapelinhalt zeigt der Befehl `dirs` an.

- `echo` wurde in der Bash ebenfalls erweitert: Soll der Zeilenumbruch am Ende der Ausgabe unterdrückt werden, verwendet man den Parameter `-n`. Wer die von der *C*-Funktion `printf` bekannten Ausgabesteuerzeichen verwenden will, gibt den Parameter `-e` an.

- `hash` (ohne Parameter) zeigt die Liste der Pfade zu den Programmen an. Um die Zugriffe auf Programme zu beschleunigen, verwaltet die Bash einen internen Cache der Pfade auf bereits gestartete Programme. Wird ein Programm erneut gestartet, kann die zeitaufwändige Suche entlang des Pfades entfallen. Bei der Gelegenheit wird auch gleich angezeigt, wie oft das Programm gestartet wurde. Wer am Abend wissen will, womit er sich den ganzen Tag über beschäftigt hat, kann mit diesem Kommando zumindest Hinweise bekommen (oder er wirft einen Blick auf die `.bash_history`). `hash -r` verwirft alle gespeicherten Pfade.

- `help` ist die Hilfefunktion der Bash. Ohne Parameter aufgerufen, listet sie alle internen Kommandos auf. Mit einem Parameter wird ein Hilfetext über den angegebenen Befehl ausgeben.
- `logout` führt das Skript `.bash-logout` aus und beendet die Login-Shell.

4.6.5 Zeichenkettenbildung mit geschweiften Klammern

`bash` setzt aus Zeichenketten, die in geschweiften Klammern angegeben werden, alle denkbaren Zeichenkettenkombinationen zusammen. Die offizielle Bezeichnung für diesen Substitutionsmechanismus lautet „Klammererweiterung" (brace expansion). Beispielsweise wird `teil{1,2,3}` zu `teil1 teil2 teil3`. Klammererweiterungen können den Tippaufwand beim Zugriff auf mehrere ähnliche Dateinamen oder Verzeichnisse reduzieren. Gegenüber Jokerzeichen wie *** und *?* haben Sie den Vorteil, dass auch noch nicht existierende Dateinamen gebildet werden können (etwa für `mkdir`).

```
user$ echo {a,b}{1,2,3}
a1 a2 a3 b1 b2 b3

user$ echo {a,b,c}{e,f,g}.{1,2,3}
ae.1 ae.2 ae.3 af.1 af.2 af.3 ag.1 ag.2 ag.3 be.1 be.2 be.3
bf.1 bf.2 bf.3 bg.1 bg.2 bg.3 ce.1 ce.2 ce.3 cf.1 cf.2 cf.3
cg.1 cg.2 cg.3
```

4.6.6 Berechnung arithmetischer Ausdrücke in eckigen Klammern

`bash` ist normalerweise nicht in der Lage, Berechnungen auszuführen. Wenn Sie *2+3* eingeben, weiß die Shell nicht, was sie mit diesem Ausdruck anfangen soll. Wenn Sie innerhalb der Shell eine Berechnung ausführen möchten, müssen Sie den Ausdruck in eckige Klammern setzen und ein $-Zeichen voranstellen.

```
user$ echo $[2+3]
5
```

Innerhalb der eckigen Klammern sind die meisten aus der Programmiersprache C bekannten Operatoren erlaubt: `+ - * /` für die vier Grundrechenarten, `%` für Modulo-Berechnungen, `== != < <= >` und `>=` für Vergleiche, `<<` und `>>` für Bitverschiebungen, `! &&` und `||` für logisches NICHT, UND und ODER etc. Alle Berechnungen werden für 32-Bit-Integerzahlen ausgeführt (Zahlenbereich zwischen +/-2147483648). Wenn einzelne Werte aus Variablen entnommen werden sollen, muss ein `$`-Zeichen vorangestellt werden. Eine alternative Möglichkeit, Berechnungen durchzuführen, bietet das Kommando `expr`. Dabei handelt es sich um ein eigenständiges Linux-Kommando, das unabhängig von `bash` funktioniert.

4.6.7 Ausgabevervielfachung mit tee

Gelegentlich kommt es vor, dass die Ausgaben eines Programms zwar in einer Datei gespeichert werden sollen, dass aber dennoch (parallel) am Bildschirm der Programmverlauf verfolgt werden soll. In diesem Fall ist eine Verdoppelung der Ausgabe erforderlich, wobei eine Kopie am Bildschirm angezeigt und die zweite Kopie in einer Datei gespeichert wird. Diese Aufgabe übernimmt das Kommando `tee`:

`user$` **ls | tee inhalt**

Das Inhaltsverzeichnis des aktuellen Verzeichnisses wird am Bildschirm angezeigt und gleichzeitig in der Datei `inhalt` gespeichert. Dabei erfolgt zuerst eine Weiterleitung der Standardausgabe an das Kommando `tee`. Dieses Kommando zeigt standardmäßig die Standardausgabe am Terminal an und speichert die Kopie davon in der angegebenen Datei. Dass es sich wirklich um eine Vervielfachung der Ausgabe handelt, bemerken Sie, wenn Sie auch die Standardausgabe von `tee` in eine Datei weiterleiten:

`user$` **ls | tee inhalt1 > inhalt2**

Das Ergebnis sind zwei identische Dateien `inhalt1` und `inhalt2`. Das obige Kommando hat reinen Beispielcharakter. Etwas schwieriger zu verstehen, dafür aber sinnvoller ist das folgende Beispiel:

`user$` **ls -l | tee inhalt1 | sort +4 > inhalt2**

In `inhalt1` befindet sich wiederum das „normale“ Inhaltsverzeichnis, das von `ls` automatisch nach Dateinamen sortiert wurde. Die Kopie dieser Ausgabe wurde an `sort` weitergegeben, dort nach der Dateigröße (fünfte Spalte, also Option +4) sortiert und in `inhalt2` gespeichert.

Übersicht Kommando-Ausführung

`kommando1; kommando2`	Führt die Kommandos nacheinander aus
`kommando1 && kommando2`	Führt K. 2 aus, wenn K. 1 erfolgreich war
`kommando1 || kommando2`	Führt K. 2 aus, wenn K. 1 einen Fehler liefert
`kommando &`	Startet das Kommando im Hintergrund
`kommando1 & kommando2`	Startet K. 1 im Hintergrund, K. 2 im Vordergrund
`(kommando1 ; kommando2)`	Führt beide Kommandos in der gleichen Shell aus

Übersicht Substitutionsmechanismen

Jokerzeichen für Dateinamen

`?`	Genau ein beliebiges Zeichen
`*`	Beliebig viele (auch null) beliebige Zeichen

	(aber keine .*-Dateien!)
`[abc]`	Eines der angegebenen Zeichen
`[a-f]`	Ein Zeichen aus dem angegebenen Bereich
`[!abc]`	Keines der angegebenen Zeichen
`[^abc]`	Wie oben
`~`	Abkürzung für das Heimatverzeichnis
`.`	Aktuelles Verzeichnis
`..`	Übergeordnetes Verzeichnis

Zeichenkettenzusammensetzungen

`ab{1,2,3}`	Liefert `ab1 ab2 ab3`

Arithmetik

`$[3*4]`	Arithmetische Berechnungen

Kommandosubstitution

`` `kommando` ``	Ersetzt das Kommando durch sein Ergebnis
`$(kommando)`	Wie oben, alternative Schreibweise

Auswertung von Zeichenketten

`kommando "zeichen"`	Verhindert die Auswertung von Sonderzeichen
`kommando 'zeichen'`	Wie oben, aber noch restriktiver

4.7 Reguläre Ausdrücke, *grep* und *sed*

4.7.1 Reguläre Ausdrücke (*Regular Expressions*)

Vielfach werden in Shell-Skripten Textdateien manipuliert oder Textzeilen nach bestimmten Zeichenfolgen durchsucht. In solchen Fällen hilft das Kommando `grep` (global regular expression print). So liefert das folgende Kommando eine Liste aller Prozesse, welche die Zeichenkette „apache“ enthalten:

```
user$  ps ax | grep "apache"
```

`grep` sucht also nach einer Zeichenkette im Eingabedatenstrom. Aber das, was da als Suchbegriff stehen kann, geht weit über eine einfache Zeichenkette hinaus – es sind *Textmuster*, so genannte *reguläre Ausdrücke* erlaubt. Worum handelt es sich dabei?

Beim Suchen (und Ersetzen) von Textmustern steht hiermit ein mächtiges Werkzeug zur Verfügung. Es wirkt ein ähnlicher Mechanismus, wie er bereits bei der Auswahl von Dateien in der Shell besprochen wurde – die Möglichkeiten gehen aber sehr viel weiter. Die einfachste Form ist eine einfache Zeichenkette als Suchmuster. Diese Zeichenkette kann spezielle Zeichen (Metazeichen) mit besonderer Bedeutung enthalten. Wichtig sind hier vor allem folgende Metazeichen:

- `^` steht für den Zeilenbeginn
  `/^Meier/` adressiert im ex-Modus des `vi` die Zeile, die mit `Meier` beginnt
- `$` steht für das Zeilenende,
  `/Meier$/` adressiert im ex-Modus des `vi` die Zeile, die mit `Meier` endet.

- `[...]` definiert *einen* Buchstaben aus der Zeichenmenge in `[...]`. Beginnt die Aufzählung der Zeichen mit `^`, wird nach einem Zeichen gesucht, das *nicht* in der Menge enthalten ist (bei den Dateinamen war dies das `!`-Zeichen).
  `[ABC]`: einer der Buchstaben `A`, `B` oder `C`
  `[A-Z]`: Großbuchstaben
  `[A-Za-z]`: alle Buchstaben
  `[^0-9]`: keine Ziffer
  Die Zeichen „]“, „-“ und „^“ benötigen eine Sonderbehandlung, wenn sie als „normales“ Zeichen in einer Menge vorkommen sollen. Mengenangaben bei regulären Ausdrücken können somit enthalten:
  - Zeichen dicht geschrieben: `[abcx123]`
  - Bereich von ... bis ...: `[a-fu-x]` (alle Zeichen, deren ASCII-Code dazwischen liegt)
  - das Zeichen „^“ (*nicht* als erstes): `[abx^17]`
  - das Zeichen „]“ (*nur* als erstes): `[]abx17]`
  - das Zeichen „-“ (*nur* als erstes oder letztes): `[-+.0-9]`
- `.` der Punkt steht für ein beliebiges Zeichen (wie `?` in der Shell).
- `*` steht für eine beliebige Folge des *vorhergehenden* Zeichens (auch gar kein Zeichen!).
  `a*`: Leerstring oder beliebige Folge von `a`’s
  `aa*`: eine beliebige Folge von `a`’s (mindestens eines)
  `[a-z]*`: Leerstring oder eine beliebige Folge von Kleinbuchstaben
  `[a-z][a-z]*`: eine beliebige Folge von Kleinbuchstaben (mindestens einer)
  `.*`: jede beliebige Zeichenfolge
- `+` steht wie `*` für eine beliebige Folge des *vorhergehenden* Zeichens, jedoch muss das Zeichen mindestens einmal auftauchen.
  `a+` steht für eine beliebige Folge von `a`’s, jedoch mindestens eines (entspricht `aa*`).
- \ hebt den Metazeichen-Charakter für das folgende Zeichen auf.
  `a*` steht für Leerstring oder beliebige Folge von `a`’s.
  `a\*` steht für die Zeichenfolge `a*`.
- `\(...\)` Reguläre Ausdrücke können mit Klammern gruppiert werden. Damit die Klammern nicht als Teil der Zeichenkette aufgefasst werden, muss ein \ davor stehen.

\([A-Za-z]*\) gruppiert beispielsweise ein Wort aus beliebig vielen Buchstaben, wobei auch ein leeres Wort (0 Buchstaben) dazu gehört. Soll das Wort mindestens einen Buchstaben enthalten, muss man \([A-Za-z][A-Za-z]*\) schreiben. Die Anwendung solcher Gruppen wird weiter unten gezeigt – es kann in den Editor-Befehlen nämlich Bezug auf die Gruppen genommen werden.

- `\i` Referenzieren des `i`-ten Klammerausdrucks (siehe Beispiele weiter unten)

Reguläre Ausdrücke sind in doppelter Hinsicht von Bedeutung: zur Adressierung von Zeilen und beim Ersetzen von Zeichenketten. Mit regulären Ausdrücken lässt sich der Ersetzungsbefehl erweitert verwenden. Reguläre Ausdrücke basieren auf einem nicht-deterministischen finiten Automaten, der folgendermaßen vorgeht: Er merkt sich die Stellen, an denen mehr als eine Möglichkeit zu kontrollieren ist. Stellt er beim Testen einer Variante fest, dass der Gesamtausdruck nicht mehr zutrifft, geht er zurück zum „Scheideweg“ und prüft die Alternative. Erst wenn alle abgehakt sind, entscheidet der NFA, ob der Ausdruck zutrifft oder nicht. Durch dieses „Backtracking“ genannte Vorgehen beherrscht ein Programm (z. B. die Editoren `vi` oder `sed` nummerierte Rückbezüge wie in `s/(Eins) (Zwei)/\2 \1/g`. Hier sorgen die Klammern dafür, dass sich der Editor jedes „Eins“ und jedes „Zwei“ merkt. Im zweiten Teil vertauscht dann `\2 \1` die beiden miteinander.

In den folgenden Beispielen wird von einer Telefonliste ausgegangen, die aus Namen, Vornamen und Telefonnummern besteht:

```
Huber Karl      123
Meier Hans      231
Gaukeley Gundel 781
Schulze Maria   256
Weber Klaus     400
```

Für die Befehle in den Beispielen soll uns der Editor `vi` im Zeilenediter-Modus (ex-Modus) dienen. Bei der Shell-Programmierung kommt natürlich kein interaktiver Editor wie `emacs`, `vi` oder `ed` zum Einsatz (obwohl das auch ginge), sondern der unten beschriebene `sed`, der das/die Editierkommando(s) als alle Zeilen in der Standardeingabe anwendet und das Ergebnis in die Standardausgabe schreibt. Die Zeilen beginnen immer mit „1,$s“, was bedeutet: „suche und ersetze in allen Zeilen des Textes“. Die Telefonnummern sollen um den Text `Tel.:` ergänzt werden. Dazu wird ein weiteres Feature der `s`-Anweisung verwendet, der oben beschriebene Rückbezug.

```
1,$s/\([0-9][0-9]*\)\$/Tel.: \1/
```

Das Ergebnis sieht dann folgendermaßen aus:

```
Huber Karl      Tel.: 123
Meier Hans      Tel.: 231
```

```
Gaukeley Gundel Tel.: 781
Schulze Maria   Tel.: 256
Weber Klaus     Tel.: 400
```

Jetzt sollen Nachname und Vorname vertauscht werden (beachten Sie die Leerzeichen zwischen den Gruppen):

```
1,$s/\{[A-Za-z][A-Za-z]*\) \{[A-Za-z][A-Za-z]*\)/\2/ \1/
```

Das Ergebnis:

```
Karl Huber      Tel.: 123
Hans Meier      Tel.: 231
Gundel Gaukeley Tel.: 781
Maria Schulze   Tel.: 256
Klaus Weber     Tel.: 400
```

Hinweis

Aus Gründen der Übersichtlichkeit wurden die Umlaute und das scharfe S weggelassen – in einer realen Anwendung müssen die natürlich mit in die eckige Klammer. Gegebenenfalls sind auch noch weitere Buchstaben zu berücksichtigen, etwa die Vokale mit Akzent.

In einem letzten Schritt werden die Telefonnummern aus der Liste entfernt:

```
1,$s/([0-9][0-9]*\$//
```

Das Ergebnis:

```
Karl Huber
Hans Meier
Gundel Gaukeley
Maria Schulze
Klaus Weber
```

Reguläre Ausdrücke versuchen normalerweise, den frühesten Treffer im String zu finden. Kommt aber ein Quantifizierer wie `*` ins Spiel, will der reguläre Ausdruck so viel wie möglich finden, er wird gierig („greedy“). Dabei ist die „Gierigkeit“ stärker als die „Links-Bindung“.
Will man beispielsweise in HTML-Code einen bestimmten Tag erwischen, heisst der erste Versuch vermutlich: `/<.*>/`. übersetzt: „Suche beliebig viele (auch gar kein) Zeichen, umschlossen von spitzen Klammern.“ Was würde Perl nun in der Zeile

```
<B>Wir</B> sind die <B>Champions</B>!
```

finden? Alles von der ersten spitzen Klammer bis zur letzen vor dem Ausrufezeichen. Da ein Quantifizierer dabei ist, gilt nicht mehr „Treffer soweit links wie möglich“ (also das erste `<B>`) sondern „So viel wie möglich“.

Die Gierigkeit lässt sich jedoch durch ein hinter + oder * gesetztes Fragezeichen beschränken. Benutzt man im obigen Beispiel <.*?>, wird es <B> finden. In solchen Fällen hilft ebenfalls: /<[^>]+>/. Dieser Ausdruck sucht ein <, dann etwas, was kein > ist, davon mindestens eines, schließlich ein >.

Es gibt ein paar Standard-Tools, die reguläre Ausdrücke verwenden. Das bekannteste ist das schon erwähnte Text-Suchprogramm grep.

4.7.2 *grep*

Der Aufruf erfolgt über das Kommando

grep *reg. Ausdruck* [*Optionen*] [*Dateiname(n)*]

grep steht für „global regular expression print“. Das Kommando ist ein Tool zum Durchsuchen von Dateien auf Strings, die mit regulären Ausdrücken definiert sind. Die gefundenen Zeilen werden auf die Standardausgabe ausgegeben. Wegen der Angabe des regulären Ausdrucks in der Befehlszeile muss der reguläre Ausdruck in " ... " oder ´ ... ´ eingeschlossen werden, wenn er Leerzeichen oder Sonderzeichen enthält, die von der Shell ersetzt werden. Wurde keine Datei angegeben, liest *grep* von der Standardeingabe. Daneben gibt es einige nützliche Optionen:

-q Nur Return-Wert zurückgeben (gefunden/nicht gefunden)

-n Zeilennummern mit ausgeben

-c Nur die Anzahl der Zeilen ausgeben, für die der reguläre Ausdruck erfüllt ist

-l Nur die Namen der Dateien ausgeben, in denen etwas gefunden wurde

-L Nur die Namen der Dateien ausgeben, in denen *nichts* gefunden wurde

-i Groß-/Kleinschreibung nicht unterscheiden

-v Alle Zeilen ausgeben, in denen der reguläre Ausdruck *nicht* erfüllt ist.

Daneben kennt grep noch etliche andere Optionen, die aber wesentlich seltener gebraucht werden. Hier einige Beispiele:

- Suchen nach allen „Meiers“, „Maiers“, „Meyers“, „Mayers“ in einer Datei:

  user$ **grep 'M[ae][iy]er' Namen**

- Suchen nach allen Zeilen, die mit dem Wort „Beispiel“ beginnen; Ausgabe der Zeilennummern:

  user$ **grep -n '^Beispiel' Vorlesung**

- Wichtig ist *grep* für den Systemverwalter, z. B. bei der Suche nach Login-Namen in der Benutzerdatei, damit nicht derselbe Name zweimal verwendet wird:

  `user$` **grep '^klaus:' /etc/passwd**

  Bei *grep* wird auch vielfach nur der Rückgabewert des Programms (nicht die Ausgabe!) verwendet. In Skript-Konstruktionen (siehe später) kann das beispielsweise so aussehen:

  `user$` **grep -q '^klaus:' /etc/passwd && echo "Gibts schon"**

4.7.3 Der Stream-Editor *sed*

sed liest aus der angegebenen Eingabedatei (normalerweise Standardeingabe) Zeile für Zeile in seinen Eingabepuffer, führt die Editieranweisungen auf die Zeile aus und schreibt sie auf die Standardausgabe. *sed* beendet seine Arbeit, wenn das Ende der Eingabedatei erreicht ist oder explizit eine Beendigungsanweisung erreicht wurde. Ein zweiter Puffer, der Haltepuffer, dient zum Zwischenspeichern von Ergebnissen. Aufruf:

sed [-n] [-e '*sed-Befehle*'] [-f *Skriptdatei*] [*Eingabedatei(en)*]

Der *sed* ist ein nicht interaktiver Editor zur Dateibearbeitung. Daher muss die Befehlsfolge zum Editieren bereits beim Aufruf festliegen. Der Inhalt der Eingabedatei(en) oder der Standardeingabe (wenn keine Eingabedatei angegeben wurde) wird durch die Editieranweisungen modifiziert und dann auf die Standardausgabe ausgegeben. Mit der Option `-f` *Skriptdatei* entnimmt der *sed* die Editieranweisungen dieser Skriptdatei. Beachten Sie, dass die Eingabedatei selbst nicht verändert wird! Soll das Ergebnis der *sed*-Anweisung in einer (anderen) Datei gespeichert werden, muss die Standardausgabe in diese Datei umgelenkt werden. Optionen:

`-n` Meldungen werden unterdrückt (sinnvoll im Zusammenhang mit Pipes, die in den `sed` hineingeleitet werden).

`-e` Es werden die auf die Option `-e` folgenden Anweisungen für die Bearbeitung der Eingabedatei verwendet. Bei mehr als einer Anweisung müssen die Anweisungen durch Semikola voneinander getrennt werden. Das Semikolon muss dabei ohne Leerzeichen direkt hinter der Anweisung stehen! Diese Option kann mehrfach angegeben werden.
Um Fehlinterpretationen der Shell zu vermeiden, sollte die Editieranweisung grundsätzlich in Hochkommata oder „Gänsefüßchen“ eingeschlossen werden.

`-f` Wird diese Option angegeben, liest *sed* seine Editieranweisungen aus der angegebenen Datei. Die Anweisungen müssen entweder durch Semikola getrennt werden oder jede Anweisung muss in einer eigenen Zeile stehen. Leerzeichen hinter einer Anweisung sind nicht erlaubt. Die Anweisungen dürfen nicht in Hochkommata eingeschlossen werden. Durch mehrfache Angabe der Option können mehrere Skriptdateien zugewiesen werden.

`-r` Erweiterte reguläre Ausdrücke verwenden.

`sed` kennt dieselben Anweisungen wie der interaktive Editor `ed` oder der `vi` im ex-Modus. Auch die Adressierung (Zeilennummern, reguläre Ausdrücke) erfolgt bei allen auf die gleiche Weise. Editieranweisungen haben die folgende Form:

```
[Adresse1 [,Adresse2]] Anweisung [Argumente]
```

Für jede Zeile der Eingabedatei werden alle Anweisungen ausgeführt. Durch Angabe einer Zeile (*Adresse1*) oder eines Zeilenbereichs (*Adresse1, Adresse2*) kann man deren Wirkung jedoch einschränken. Anstelle von Zeilennummern können auch reguläre Ausdrücke verwendet werden:

```
[/Muster/] Anweisung [Argumente]
[/Muster1/] [/Muster2/] Anweisung [Argumente]
```

Im ersten Fall wird jede Zeile, die die Zeichenkette *Muster* enthält, von der Anweisung bearbeitet, im zweiten Fall alle Bereiche der Eingabedatei, die mit einer Zeile, die *Muster1* enthält, beginnen und mit einer Zeile, die *Muster2* enthält, enden.

Zusätzlich hat der *sed* noch weitere Funktionen (z. B. Test- und Sprungfunktionen, Klammerung, Wiederholungen). Er eignet sich daher besonders für Shell-Programme. Interessant sind beim *sed* die Klammern:

`( )` Die runden Klammern müssen, wie schon erwähnt, mit \ geschützt werden. Sie legen die Gruppierung der einzelnen Komponenten fest.

{ } Die geschweiften Klammern müssen ebenfalls mit \ geschützt werden. Sie legen Wiederholungen fest. Im Folgenden stehen `Z` für ein Zeichen und `M` und `N` für Zahlen zwischen 0 und 253.

1. genau `M` Wiederholungen: `Z\{M\}`; z. B. alle Namen mit genau 8 Zeichen Länge: `sed -n '/^.\{8\}/p' namen`
2. mindestens `M` Wiederholungen: `Z\{M, \}`
3. zwischen `N` und `M` Wiederholungen: `Z\{N,M\}`

Am häufigsten kommt die Zeichenketten-Ersetzungsanweisung in Shell-Anwendungen vor. Dazu ein etwas komplexeres Beispiel: Es soll einer Datei mit der Endung `.tex` auf eine Datei gleichen Namens, jedoch mit der Endung `.bak`, kopiert werden. Aus `diplom-kap1.tex` wird beispielsweise `diplom-kap1.bak`. Den Dateinamen *ohne* Endung übergeben wir mal in einer Shell-Variablen:

1. Erst einmal probieren wir die Generierung des neuen Namens. Da `sed` seine Daten auf der Standardeingabe erwartet, pipen wir mittels `echo` den Variableninhalt hinein:

```
user$  DATEI="diplom-kap1.tex"
user$  echo $DATEI | sed -e 's/ṫex/.bak/'
diplom-kap1.bak
```

   Da der Punkt ein Metazeichen ist, musste er im ersten Ausdruck, der ja ein regulärer Ausdruck ist, geschützt werden.

2. Jetzt muss der neue Name irgendwie zum Kopieren verwendet werden. Deshalb nützt er auf der Standardausgabe relativ wenig. Aber wir haben ja eine Möglichkeit, den Output wieder in eine Variable zu packen:

```
user$  DATEI="diplom-kap1.tex"
user$  NEU=` echo $DATEI | sed -e 's/ṫex/.bak/'`
user$  echo $NEU
diplom-kap1.bak
```

3. Das hat also geklappt. Nun kann man das Kopierkommando anhängen:

```
user$  DATEI="diplom-kap1.tex"
user$  NEU=` echo $DATEI | sed -e 's/ṫex/.bak/'`
user$  cp $DATEI $NEU
```

4. Die Profis packen natürlich alle Aktionen in eine Zeile:

```
user$  DATEI="diplom-kap1"
user$  cp $DATEI ` echo $DATEI | sed -e 's/ṫex/.bak/'`
```

Wenn wir nun noch eine Schleife um die Zeile mit der Kopieranweisung setzen, in der die Variable `DATEI` nacheinander alle Dateinamen annimmt, die umbenannt werden sollen, haben wir schon ein nettes Tool.

Noch ein Beispiel – Umlaute HTML-gerecht ersetzen:

```
cat textfile | sed -e 's/ä/\\&auml;/g' \
                   -e 's/ö/\\&ouml;/g' \
                   -e 's/ü/\\&uuml;/g' \
                   -e 's/Ä/\\&Auml;/g' \
                   -e 's/Ö/\\&Ouml;/g' \
                   -e 's/Ü/\\&Uuml;/g' \
                   -e 's/ß/\\&szlig;/g' > htmlfile
```

Die folgende Liste fasst die wichtigsten *sed*-Kommandos zusammen:

- `a\` *text* (*append lines*)

  text wird nach der aktuellen Eingabezeile auf die Standardausgabe geschrieben. Besteht der Text aus mehreren Zeilen, so muss das Fortsetzungszeichen \ am Zeilenende vor ⏎ angegeben werden.

- `b[`*marke*`]` (*branch to label*)

  Es wird zu der *marke* (in der Form `:`*marke* angegeben) des *sed*-Skripts gesprungen und dort die Abarbeitung des Skripts fortgesetzt. Fehlt die Angabe der Marke, so wird die nächste Eingabezeile bearbeitet.

- `c\ [`*text*`]` (*change lines*)

  Der Inhalt des Eingabepuffers wird durch *text* ersetzt. Besteht *text* aus mehreren Zeilen, so muss das Fortsetzungszeichen \ am Zeilenende vor ⏎ angegeben werden.

- `d` (*delete lines*)

  Der Inhalt des Eingabepuffers wird gelöscht (nicht ausgegeben) und das *sed*-Skript wird sofort wieder von Beginn an mit dem Lesen einer neuen Eingabezeile gestartet.

- `i\` *text* (*insert lines*)

  Der *text* wird vor der aktuellen Eingabezeile auf die Standardausgabe geschrieben. Besteht *text* aus mehreren Zeilen, muss das Fortsetzungszeichen \ am Zeilenende vor ⏎ angegeben werden.

- `l` (*list pattern space on the standard output*)

  Der Inhalt des Eingabepuffers wird auf die Standardausgabe geschrieben, wobei nicht druckbare Zeichen durch ihren ASCII-Wert (2-Ziffer) ausgegeben werden. Überlange Zeilen werden als mehrere einzelne Zeilen ausgegeben.

- `n` (*next line*)

  Der Eingabepuffer wird auf die Standardausgabe ausgegeben, dann wird die nächste Eingabezeile in den Eingabepuffer gelesen.

- `N` (*Next line*)

  Die nächste Eingabezeile wird an den Eingabepuffer (mit Newline-Zeichen getrennt) angehängt; die aktuelle Zeilennummer wird hierbei weitergezählt.

- `p` (*print*)

  Der Eingabepuffer wird auf die Standardausgabe ausgegeben.

- `P` (*Print first part of the pattem space*)

  Der erste Teil des Eingabepuffers (einschließlich des ersten Newline) wird auf die Standardausgabe ausgegeben.

- `q` (*quit*)

  Nach Ausgabe des Eingabepuffers (nicht bei Option `-n`) wird zum Skriptende gesprungen und die Skriptausführung beendet.

- `r` *datei* (*read the contents of a file*)

  Es wird die Datei *datei* gelesen und ihr Inhalt auf die Standardausgabe ausgegeben, bevor die nächste Eingabezeile gelesen wird. Zwischen `r` und *datei* muss genau ein Leerzeichen sein.

- `s/`*rA*`/`*txt*`/[`*flag*`]`

  (*substitute*) Im Eingabepuffer werden die Textstücke, die durch den regulären Ausdruck *rA* abgedeckt sind, durch den String *txt* ersetzt. Anstelle des Trennzeichens `/` kann jedes beliebige Zeichen verwendet werden. Die *flags* legen fest, wie der Ersetzungsprozess durchgeführt werden soll. Unter anderem gibt es:

  `n:` Es wird nur `n`-mal in der Zeile ersetzt.

  `g` (*global*): Es werden alle passenden Textstücke, die sich nicht überlappen, ersetzt.

  Ohne *flag* wird nur das erste passende Textstück ersetzt.

- `w` *datei* (*write to a file*)

  Es wird der Eingabepuffer an das Ende der Datei *datei* geschrieben. Zwischen `w` und *datei* muss genau ein Leerzeichen sein.

- `y/`*st1*`/`*st2*`/`

  Im Eingabepuffer werden alle Zeichen, die in *st1* vorkommen, durch die an gleicher Stelle in *st2* stehenden Zeichen ersetzt. *st1* und *st2* müssen gleich lang sein.

Die folgende Übersicht in Tabelle 4.2 fasst die verschiedenen Möglichkeiten zusammen.

Tabelle 4.2: Metazeichen bei regulären Ausdrücken

Ausdruck	Beschreibung
ˆ	steht für den Zeilenbeginn qqˆMeier adressiert eine Zeile, die mit „Meier“ beginnt
$	steht für das Zeilenende „Meier$“ adressiert Zeile, die mit „Meier“ endet
`[ ]`	definiert einen Buchstaben aus dem Bereich in `[ ]` `[ABC]`: einer der Buchstaben A, B oder C `[A-Z]`: Großbuchstaben `[A-Za-z]`: alle Buchstaben Beginnt der Bereich mit ˆ, wird nach einen Zeichen gesucht, das **nicht** im Bereich enthalten ist. `[^0-9]`: keine Ziffer
.	Der Punkt steht für ein beliebiges Zeichen.
*	Der * steht für eine beliebige Folge des **davorstehenden** Zeichens (auch 0 Zeichen!). `a*`: Leerstring oder beliebige Folge von „a“ `aa*`: eine beliebige Folge von „a“ (min. 1) `[a-z]*`: Leerstring oder Folge von Kleinbuchstaben `.*`: jede beliebige Zeichenfolge
*?	Der Stern „*“ ist recht „gefräßig“ (greedy), d. h. es wird versucht, maximal viele Zeichen in den regulären Ausdruck einzuschließen. Bei der Zeichenkette „aaa:bbb:ccc“ würde der Ausdruck „.*:“ den String „aaa:bbb:“ finden. Durch das nachgestellte Fragezeichen wird diese Eigenschaft umgekehrt, es wird die minimale Teilzeichenkette genommen, also im obigen Beispiel „`aaa:`“.
+	Das + steht für eine beliebige Folge des **davorstehenden** Zeichens, jedoch mindestens eines. `a+`: a, aa, aaa, aaaa, ...
?	Nullmal oder einmal das **davorstehende** Zeichen
\	Metazeichen-Charakter des folgenden Zeichens aufheben `a*` steht für Leerstring oder beliebige Folge von a’s. `a\*` steht für die Zeichenfolge „a*“.
(...)	Gruppieren regulärer Ausdrücke `([A-Za-z]*)` gruppiert beispielsweise ein Wort aus beliebig vielen Buchstaben, wobei auch ein leeres Wort (0 Buchstaben) dazugehört. Soll das Wort mindestes einen Buchstaben enthalten, muss man `([A-Za-z][A-Za-z]*)` oder `([A-Za-z]+)` schreiben.
\i	Referenzieren des i-ten Klammerausdrucks
&	Referenzieren des Suchausdrucks (beim s-Befehl) `s/hallo/& &/` erzeugt „hallo hallo“

4.8 Shell-Programmierung

Die Shell dient nicht nur der Kommunikation mit dem Anwender, sondern sie kennt die meisten Konstukte einer Programmiersprache. Es lassen sich Anweisungen in einer Textdatei speichern, die dann wie ein beliebiges anderes Unix-Kommando aufgerufen werden kann. Solche Dateien nennt man *Shell-Skripten* oder *shell scripts*. Ein Shell-Skript kann aufgerufen werden über eine Subshell (`sh` *Dateiname*) oder über den Namen, wenn die Excecute-Berechtigung gesetzt ist (mittels des Befehls `chmod` *Dateiname*).

Das Shell-Skript wird mit einem Editor erstellt und kann alle Möglichkeiten der Shell nutzen, die auch bei der interaktiven Eingabe möglich sind. Insbesondere kann auch die Umleitung der Ein-/Ausgabe wie bei einem Binärprogramm erfolgen. Selbstverständlich lassen sich auch innerhalb eines Skripts weitere Shell-Skripten aufrufen. Dem Shell-Skript können Parameter übergeben werden; es ist damit universeller verwendbar.

4.8.1 Testen von Shell-Skripten

Die Ersetzungsmechanismen der Shell machen es manchmal nicht leicht, auf Anhieb korrekt funktionierende Skripten zu erstellen. Zum Testen bieten sich daher einige Möglichkeiten an:

- Einfügen von `echo`-Kommandos anstelle der vorgesehenen Kommandos:

  `echo` [*Argumente*]

  Dieses Kommando gibt die Argumente auf dem Bildschirm aus. Für den Einsteiger ist das Kommando wichtig, weil er so die Kommando-Bearbeitung der Shell recht gut verfolgen und studieren kann.

- Aufruf über Subshell mit Optionen:

  `-v` (*verbose*) Die Shell gibt alle bearbeiteten Befehle aus.

  `-x` (*execute*) Die Shell gibt die Ersetzungen aus.

  `-n` (*noexecute*) Die Shell gibt die Befehle zwar aus, sie werden jedoch nicht ausgeführt.

  Typischer Kommando-Aufruf: `sh -vx` *Dateiname*

- „Kritische“ Kommandos (z. B. `rm`) sollten Sie zunächst durch vorangestellte `echo`-Befehle „entschärfen“.

- Ein Doppelpunkt vor einer Zeile wirkt wie ein Kommentar: Das Kommando wird nicht ausgeführt, aber die Parametersubstitution erfolgt.

4.8.2 Kommentare in Shell-Skripten

Wie Programme müssen auch Shell-Skripten kommentiert werden. Kommentare werden durch das Zeichen `#` eingeleitet. Alles was in einer Zeile hinter dem `#` steht, wird als Kommentar betrachtet (übrigens betrachten auch nahezu alle anderen Unix-Programme das `#` als Kommentarzeichen in Steuer- und Parameterdateien). Leer- und Tabulatorzeichen können normalerweise in beliebiger Anzahl verwendet werden. Da die Shell Strukturen höherer Programmiersprachen enthält, ist durch Einrücken eine übersichtliche Gestaltung der *scripts* möglich. Auch eingestreute Leerzeilen sind nahezu überall erlaubt.

Bei vielen Skripten findet man eine Sonderform der Kommentarzeile zu Beginn, die beispielsweise so aussieht:

```
#!/bin/sh
```

Durch diesen Kommentar wird festgelegt, welches Programm für die Ausführung des Skripts verwendet wird. Er wird hauptsächlich bei Skripten verwendet, die allen Benutzern zur Verfügung stehen sollen. Da möglicherweise unterschiedliche Shells verwendet werden, kann es je nach Shell (`sh`, `csh`, `ksh`,...) zu Syntaxfehlern bei der Ausführung des Skripts kommen. Durch die Angabe der ausführenden Shell wird dafür gesorgt, dass nur die „richtige“ Shell verwendet wird. Die Festlegung der Shell stellt außerdem einen Sicherheitsmechanismus dar, denn es könnte ja auch ein Benutzer eine modifizierte Shell verwenden. Neben den Shells können auch andere Programme zur Ausführung des Skripts herangezogen werden; häufig sind *awk*- oder *Perl*-Skripten.

In Shell-Script-Dateien dürfen die Zeilen auch nicht durch die Windows-typische Kombination aus Carriage Return und Linefeed getrennt sein. Das kann z. B. passieren, wenn die Dateien unter Windows erstellt und dann nach Linux kopiert wurden. In diesem Fall liefert `bash` die Fehlermeldung *bad interpreter*, weil eben das Carriage Return am Zeilenende Teil des Dateinamens ist und es ein Programm „`bash^M`“ nicht gibt (das kann ihnen übrigens auch mit awk- oder perl-Programmen passieren). Die Carrage Returns bekommt man mit `vi` oder `recode` weg.

Hinweis

Shell-Script-Programme können nur ausgeführt werden, wenn die Zugriffsbits für den Lesezugriff (r) und die Ausführung (x) gesetzt sind (`chmod a+rx datei`). In der ersten Zeile eines Scripts dürfen keine deutschen Sonderzeichen verwendet werden, auch nicht in Kommentaren. Die `bash` weigert sich sonst, die Datei auszuführen.

4.8.3 Shell-Variable

Variablen sind frei wählbare Bezeichner (Namen), die beliebige Zeichenketten aufnehmen können. Bestehen die Zeichenketten nur aus Ziffern, werden sie von

bestimmten Kommandos als Integer-Zahlen interpretiert (z. B. `expr`). Bei Variablen der Shell sind einige Besonderheiten gegenüber anderen Programmiersprachen zu beachten. Im Umgang mit Variablen lassen sich grundlegend drei Formen unterscheiden: Die Variablendeklaration (z. B. `$foo=`), die Wertzuweisung (z. B. `$foo=bar`) und die Wertreferenzierung (z. B. `$echo$foo`).

Im Allgemeinen werden Variablen in der Shell nicht explizit deklariert. Vielmehr ist in der Wertzuweisung die Variablendeklaration implizit enthalten. Wird eine Variable dennoch ohne Wertzuweisung deklariert, so wird bei der Wertreferenzierung ein leerer String zurückgegeben.

Für Variable gilt allgemein:

- Variablen sind frei wählbare Bezeichner (Namen), die mit einem Buchstaben beginnen und bis zu 200 Zeichen lang sein dürfen. Leerzeichen innerhalb von Variablen sind (normalerweise) nicht erlaubt (und schlechter Stil).
- Ist der Wert, d. h der Inhalt einer Variablen gemeint, wird ein `$`-Zeichen vor den Namen gestellt (s. o.).
- Mittels spezieller Funktionen kann eine Variable auch numerisch oder logisch interpretiert werden.
- Shell-Skripten dürfen ihrerseits wieder Shell-Skripten oder Programme aufrufen. Dabei muss die Vererbung einer Variablen ausdrücklich festgelegt werden (exportieren durch den `export`-Befehl), andernfalls kann eine andere Shell – oder ein beliebiges anderes Programm – nicht darauf zugreifen.
- Jede Subshell läuft in einer eigenen Umgebung, d. h., Variablendefinitionen (und die Wirkung verschiedener Kommandos) in der Subshell sind nach deren Beendigung wieder „vergessen". Es ist nicht möglich, den Wert einer Variablen aus einem Unterprogramm in die aufrufende Ebene zu übergeben.
- Die Zuweisung eines Wertes an die Variable erfolgt mittels Gleichheitszeichen:
  - `VAR=`*Wert*

    Wert ist ein String (ohne Leerzeichen und Sonderzeichen).
  - `VAR="`*Wert*`"`

    Wert ist ein String (Ersetzung eingeschränkt, Leerzeichen und Sonderzeichen dürfen enthalten sein).
  - `` VAR=` ``*kommando*`` ` `` oder `VAR=$(`*kommando*`)`

    Wert der Variablen ist die Ausgabe des Kommandos (Newline wird zu Leerzeichen).
- Es hat sich die Konvention eingebürgert, Variablen zur Unterscheidung von Kommandos groß zu schreiben.

- Soll der Wert der Variablen mit einem String konkateniert werden, ist der Name in geschweifte Klammern einzuschließen, damit die Shell erkennen kann, wo der Variablenname endet. Bei der Zuweisung von Zahlen an Shell-Variable werden führende Leerzeichen und führende Nullen ignoriert.

Beispiele (Kommando-Eingaben beginnen mit `user$`):

```
user$ VAR="Hello World!"
user$ echo $VAR
Hello World!
```

```
user$ VAR=` pwd`
user$ echo "aktuelles Verzeichnis: $VAR"
aktuelles Verzeichnis: /home/user
user$ echo ${VAR}/bin
/home/user/bin
```

```
user$ VAR=/usr/tmp/mytmp
user$ ls > $VAR
```

Das letzte Beispiel schreibt die Ausgabe von `ls` in die Datei `/usr/tmp/mytmp`.

Enthält eine Variable ein Kommando, so kann dieses Kommando durch Angabe der Variablen ausgeführt werden, z. B.:

```
user$ VAR="echo Hallo"
user$ $VAR
Hallo
```

Innerhalb der Anführungszeichen werden Variablen unterschiedlich behandelt. Angenommen, die Variable `VAR` hat die Zeichenkette „abcdef“ zum Inhalt. Dann gilt:

- Einfache Anführungszeichen entwerten alle zwischenliegenden Metazeichen und erlauben auch keine Variablenreferenz:

  ```
  user$ echo '$VAR'
  $VAR
  ```

- Doppelte Anführungszeichen entwerten alle zwischenliegenden Metazeichen, nicht jedoch die Variablen- und Kommandosubstitution:

  ```
  user$ echo "$VAR"
  abcdef
  ```

4.8.4 Vordefinierte Variable

Beim Systemstart und beim Aufruf der Dateien `/etc/profile` (System-Voreinstellungen), `.profile` (benutzereigene Voreinstellungen) und ggf. weiteren Dateien, die ja auch Shell-Skripten sind, werden bereits einige Variablen definiert. Alle aktuell definierten Variablen können durch das Kommando `set` aufgelistet werden.

Einige vordefinierte Variablen sind neben anderen in Tabelle 4.3 aufgeführt.

Tabelle 4.3: Einige vordefinierte Shell-Variablen

Variable	Bedeutung
`HOME`	Home-Verzeichnis (absoluter Pfad)
`IFS`	Trennzeichen zwischen Parametern (in der Regel Leerzeichen, Tabulator, Newline)
`PATH`	Suchpfad für Kommandos und Skripten
`MANPATH`	Suchpfad für die Manual-Seiten
`MAIL`	Mail-Verzeichnis
`SHELL`	Name der Shell
`LOGNAME` bzw. `USER`	Login-Name des Benutzers
`PS1`	System-Prompt (`$` oder `#`)
`PS2`	Prompt für die Anforderung weiterer Eingaben (`>`)
`IFS`	(*Internal Field Separator*) Trennzeichen (meist `CR`, Leerzeichen und Tab)
`TZ`	Zeitzone (z. B. `MEZ`)

In der nächsten Tabelle 4.4 sind spezielle Variablen definiert.

Tabelle 4.4: Spezielle Shell-Variablen

Variable	Bedeutung	Kommando
`$-`	gesetzte Shell-Optionen	`set -xv`
`$$`	PID (Prozessnr.) der Shell	`kill -9 $$` („Selbstmord“)
`$!`	PID des letzten Hintergrundprozesses	`kill -9 $!` („Kindermord“)
`$?`	Exitstatus des letzten Kommandos	`cat /etc/passwd ; echo $?`

4.8.5 Parameterzugriff in Shell-Skripten

Shell-Skripten können mit Parametern aufgerufen werden, auf die über ihre Positionsnummer zugegriffen werden kann. Die Parameter können zusätzlich mit vordefinierten Werten belegt werden (später dazu mehr). Die Trennung zweier Parameter erfolgt durch die in `IFS` definierten Zeichen (Tabelle 4.5).

Tabelle 4.5: Zugriff auf Positionsparameter

Positionsparameter	Bedeutung
`$#`	Anzahl der Argumente
`$0`	Name des Kommandos
`$1`	1. Argument
.	.
.	.
.	.
`$9`	9. Argument
`$@`	alle Argumente (z. B. für Weitergabe an Subshell)
`$*`	alle Argumente konkateniert (ein einziger String)

Der Verdeutlichung soll ein kleines Shell-Skript dienen:

```
#!/bin/sh
echo "Mein Name ist $0"
echo "Mir wurden $# Parameter uebergeben"
echo "1. Parameter = $1"
echo "2. Parameter = $2"
echo "3. Parameter = $3"
echo "alle Parameter zusammen: $*"
echo "Meine Prozessnummer PID = $$"
```

Nachdem dieses Shell-Skript mit einem Editor erstellt wurde, muss es noch ausführbar gemacht werden (`chmod u+x foo`). Anschließend wird es gestartet und erzeugt die folgenden Ausgaben auf dem Bildschirm:

```
user$ ./foo eins zwei drei vier
Mein Name ist ./foo
Mir wurden 4 Parameter uebergeben
1. Parameter = eins
2. Parameter = zwei
3. Parameter = drei
Alle Parameter zusammen: eins zwei drei vier
Meine Prozessnummer PID = 3212
user$
```

Hinweis

So, wie Programme und Skripten des Unix-Systems in Verzeichnissen wie `/bin` oder `/usr/bin` gesammelt werden, ist es empfehlenswert, im Home-Directory ein Verzeichnis `bin` einzurichten, das Programme und Skripten aufnimmt. Die Variable `PATH` wird dann in der Datei `.profile` durch die Zuweisung `PATH=$PATH:$HOME/bin` erweitert. Damit die Variable `PATH` auch in Subshells (d. h. beim Aufruf von Skripten) wirksam wird, muss sie mittels `export PATH` exportiert werden.

Alle exportierten Variablen bilden das Environment für die Subshells. Informationen darüber erhält man mit den Kommandos `set` (Anzeige von Shell-Variablen) und Environment-Variablen oder `env` (Anzeige der Environment-Variablen).

In Shell-Skripten kann es sinnvoll sein, die Variablen in Anführungszeichen zu setzen, um Fehler zu verhindern (eine leere Variable erzeugt bei der Auswertung gar nichts, sind Anführungszeichen vorhanden, steht ein Leerstring da). Beim Aufruf müssen Parameter, die Sonderzeichen enthalten, ebenfalls in Anführungszeichen (am besten die einfachen) gesetzt werden. Dazu ein Beispiel. Das Skript `zeige` enthält folgende Zeile:

```
grep $1 address.dat
```

Der Aufruf `zeige 'Hans Meier'` liefert nach der Ersetzung das fehlerhafte Kommando `grep Hans Meier address.dat`, das nach dem Namen `Hans` in der Adressdatei `address.dat` und einer (vermutlich nicht vorhandenen) Datei namens `Meier` sucht. Die Änderung von `zeige` liefert bei gleichem Parameter die korrekte Version:

```
grep "$1" address.dat
```

Das Skript `zeige` soll nun enthalten:

```
echo "Die Variable XXX hat den Wert $XXX"
```

Nun wird eingegeben:

```
user$ XXX=Test
user$ zeige
Die Variable XXX hat den Wert
```

Erst wenn die Variable „exportiert“ wird, erhält man das gewünschte Ergebnis:

```
user$ XXX=Test
user$ export XXX
user$ zeige
Die Variable XXX hat den Wert Test
```

Das Skript `zeige` enthalte nun die beiden Kommandos:

```
echo "zeige wurde mit $# Parametern aufgerufen:"
echo "$*"
```

Die folgenden Kommando-Aufrufe zeigen die Behandlung unterschiedlicher Parameter:

```
user$  zeige
zeige wurde mit 0 Parametern aufgerufen:
user$  zeige eins zwei 3
zeige wurde mit 3 Parametern aufgerufen:
eins zwei 3
user$  zeige eins "Dies ist Parameter 2" drei
zeige wurde mit 3 Parametern aufgerufen:
eins Dies ist Parameter 2 drei
```

Die Definition von Variablen und Shell-Funktionen (s. u.) kann man mit `unset` wieder rückgängig machen.

4.8.6 Namens- und Parameterersetzung

Die einfache Parameterersetzung (textuelle Ersetzung durch den Wert) wurde oben gezeigt. Es gibt zusätzlich die Möglichkeit, Voreinstellungen zu vereinbaren und auf fehlende Parameter zu reagieren. Bei den folgenden Substitutionen kann bei manchen Shell-Varianten der Doppelpunkt hinter `shellvar` auch fehlen.

- `${shellvar:-`*neuerwert*`}`
  Die Variable `shellvar` ist deklariert:
  1. Die Variable `shellvar` hat einen Wert, dann wird auf diesen Wert referenziert.
  2. Die Variable `shellvar` hat keinen Wert, dann wird bei der Referenzierung der Wert *neuerwert* eingesetzt.
- `${shellvar:=`*neuerwert*`}`
  Die Variable `shellvar` ist deklariert:
  1. Die Variable `shellvar` hat einen Wert, dann wird auf diesen Wert referenziert.
  2. Die Variable `shellvar` hat keinen Wert, dann wird der Variablen `shellvar` der Wert *neuerwert* zugewiesen und bei der Referenzierung der Wert *neuerwert* eingesetzt.
- `${shellvar:?`*neuerwert*`}`
  Die Variable `shellvar` ist deklariert:
  1. Die Variable `shellvar` hat einen Wert, dann wird auf diesen Wert referenziert.

2. Die Variable `shellvar` hat keinen Wert, dann wird die Fehlermeldung *neuerwert* ausgegeben und das Shell-Skript abgebrochen.

- `${shellvar:+`*neuerwert*`}`
  Die Variable `shellvar` ist deklariert:

  1. Die Variable `shellvar` hat einen Wert, dann wird der Wert *neuerwert* referenziert.
  2. Die Variable `shellvar` hat keinen Wert, dann bleibt dieser Zustand erhalten.

- `${#shellvar}`
  liefert die Anzahl der Zeichen in der angegeben Variablen als Ergebnis (0 falls die Variable nicht existiert).

- `${shellvar#muster}`
  Vergleicht den Anfang des Variableninhalts mit dem Muster. Wird das Muster erkannt, wird der Variableninhalt abzüglich des Suchmusters zurückgegeben (kürzestmögliche Variante), andernfalls der unveränderte Variableninhalt. Im Muster dürfen die Jokerzeichen für Dateien verwendet werden.

- `${shellvar##muster}`
  Wie oben, jedoch wird die längstmögliche Variable des Musters eliminiert.

In Kommandodateien können Variablen auch Kommandonamen oder -aufrufe enthalten, da ja die Substitution vor der Ausführung erfolgt.

4.8.7 Bearbeitung einer beliebigen Anzahl von Parametern

Die Positionsparameter `$1` bis `$9` reichen nicht immer aus. Man denke nur an Skripten, die (ähnlich wie viele Kommandos) beliebig viele Dateinamen auf Parameterposition erlauben sollen. Die Shell-Skripten können mit mehr als neun Parametern versorgt werden – es wird dann mit dem Befehl `shift` gearbeitet:

```
shift
```

Eliminieren von `$1`, die weiteren Parameter werden verschoben `$2 ... $n` → `$1 ... $n-1` Dazu ein Beispiel: Das Skript `zeige` enthält folgende Befehle:

```
echo "$# Argumente:"
echo "$*"
shift
echo "Nach shift:"
echo "$# Argumente:"
echo "$*"
```

Der folgende Aufruf von `zeige` liefert:

```
user$ zeige eins zwei drei
3 Argumente:
eins zwei drei
Nach shift:
2 Argumente:
zwei drei
```

`shift` wird jedoch viel häufiger verwendet, wenn die Zahl der Parameter variabel ist. Es wird dann in einer Schleife so lange mit `shift` gearbeitet, bis die Anzahl der Parameter 0 ist. `zeige` wird nun abgeändert, um die Parameter zeilenweise auszugeben. Es läuft nun mit jeder Anzahl von Parametern:

```
while [ $# -gt 0 ]
   do
     echo $1
     shift
   done
```

4.8.8 Gültigkeit von Kommandos und Variablen

Jeder Kommandoaufruf und somit auch der Aufruf einer Befehlsdatei (Shell-Skript) hat einen neuen Prozess zur Folge. Bekanntlich wird zwar das Environment des Elternprozesses „nach unten" weitergereicht, jedoch gibt es keinen umgekehrten Weg. Auch der Effekt der Kommandos (z. B. Verzeichniswechsel) ist nur innerhalb des Kindprozesses gültig. Im Elternprozess bleibt alles beim Alten. Das gilt natürlich auch für Zuweisungen an Variablen.

Die Kommunikation mit dem Elternprozess kann aber z. B. mit Dateien erfolgen. Bei kleinen Dateien spielt sich fast immer alles im Cache, also im Arbeitsspeicher ab und ist somit nicht so ineffizient, wie es zunächst den Anschein hat. Außerdem liefert jedes Programm einen Rückgabewert, der vom übergeordneten Prozess ausgewertet werden kann. **Beachten Sie, dass sich die Shell genau umgekehrt verhält wie fast alle Programmiersprachen, bei denen ein Wert gleich 0 „falsch" und ein Wert ungleich 0 „wahr" bedeutet.**

- `= 0`: O. K.
- `> 0`: Fehlerstatus

Es gibt außerdem ein Kommando, das ein Shell-Skript in der aktuellen Shell und nicht in einer Subshell ausführt:

```
{\bf.}    (Dot) Skript ausführen
```

Das *Dot*-Kommando erlaubt die Ausführung eines Skripts in der aktuellen Shell-Umgebung, z. B. das Setzen von Variablen usw. Es funktioniert also prinzipiell wie das `include`-Makro in der Programmiersprache C.

Damit die Variable auch in Subshells (d. h. beim Aufruf von Skripten) auch wirksam wird, muss sie exportiert werden:

```
export PATH
```

Alle exportierten Variablen bilden das Environment für die Subshells.

4.8.9 Interaktive Eingaben in Shell-Skripten

Es können auch Shell-Skripten mit interaktiver Eingabe geschrieben werden, indem das `read`-Kommando verwendet wird.

`read` *variable* `[` *variable* `... ]`

`read` liest eine Zeile von der Standardeingabe und weist die einzelnen Felder den angegebenen Variablen zu. Feldtrenner sind die in `IFS` definierten Zeichen. Sind mehr Variablen als Eingabefelder definiert, werden die überzähligen Felder mit Leerstrings besetzt. Umgekehrt nimmt die letzte Variable den Rest der Zeile auf. Wird im Shell-Skript die Eingabe mit `<` aus einer Datei gelesen, bearbeitet `read` die Datei zeilenweise.

Anmerkung: Da das Shell-Skript in einer Subshell läuft, kann `IFS` im Skript umdefiniert werden, ohne dass es nachher restauriert werden muss. Die Prozedur `zeige` enthält beispielsweise folgende Befehle:

```
IFS=',' 
echo "Bitte drei Parameter, getrennt durch Kommata, eingeben:"
read A B C
echo Eingabe war: $A $B $C
```

Aufruf:

```
user$ zeige
Bitte drei Parameter, getrennt durch Komma eingeben:
eins,zwei,drei
Eingabe war: eins zwei drei
```

4.8.10 Hier-Dokumente

Die Shell bietet die Möglichkeit, Eingaben für Programme direkt in das Shell-Skript mit aufzunehmen – womit die Möglichkeit einer zusätzlichen, externen Datei wegfällt.

Eingeleitet werden so genannte Hier-Dokumente mit `<<` und einer anschließenden Zeichenfolge, die das Ende des Hier-Dokuments anzeigt. Diese Zeichenfolge steht dann alleine am Anfang einer neuen Zeile (und gehört nicht mehr zum Hier-Dokument). Bei Quoting der Ende-Zeichenfolge (eingeschlossen in Gänsefüßchen

bzw. Apostrophe) werden die Datenzeilen von den üblichen Ersetzungsmechanismen ausgeschlossen. Dazu ein Beispiel:

Das Shell-Skript `hier` enthält folgende Zeilen:

```
cat << EOT
Dieser Text wird ausgegeben, als ob er von
einer externen Datei kaeme - na ja, nicht ganz so.
Die letzte Zeile enthaelt nur das EOT und wird
nicht mit ausgegeben. Die folgende Zeile wuerde
bei der Eingabe aus einer Datei nicht ersetzt.
Parameter: $*
EOT
```

Aufruf:

```
user$  hier eins zwei
Dieser Text wird ausgegeben, als ob er von
einer externen Datei kaeme - na ja, nicht ganz so.
Die letzte Zeile enthaelt nur das EOT und wird
nicht mit ausgegeben. Die folgende Zeile wuerde
bei der Eingabe aus einer Datei nicht ersetzt.
Parameter: eins zwei
```

Außerden wäre bei der Eingabe aus einer Datei die Ersetzung von `$*` durch die aktuellen Parameter nicht möglich. Hier-Dokumente bieten also weitere Vorteile. Durch in die Kommandos eingestreute Variablen wird das Ganze variabel steuerbar. Noch ein Beispiel, diesmal die Simulation des Kommandos `wall`.

```
for  X in ‘who | cut -d’ ’ -f1‘
do
write $X << TEXTENDE
Hallo Leute,
das Wetter ist schoen. Wollen wir da nicht um 17 Uhr Schluss machen
und in den Biergarten gehen?
TEXTENDE
done
```

4.8.11 Verkettung von Kommandos

- **Hintereinanderausführung**: Will man mehrere Kommandos ausführen lassen, braucht man nicht jedes Kommando einzeln einzugeben und mit der Eingabe des nächsten Kommandos auf die Beendigung des vorhergehenden zu warten. Die Kommandos werden, getrennt durch Semikolon, hintereinander geschrieben:

  Kommando1 ; *Kommando2* ; *Kommando3*

- **Sequenzielles UND**: Das zweite Kommando wird nur dann ausgeführt, wenn das erste erfolgreich war:

  Kommando1 `&&` *Kommando2*

- **Sequenzielles EXOR**: Im Falle des Exklusiven ODER wird das zweite Kommando nur dann ausgeführt, wenn das erste erfolglos war:

  Kommando1 `||` *Kommando2*

4.8.12 Zusammenfassung von Kommandos

Kommandofolgen lassen sich – analog der Blockstruktur höherer Sprachen – logisch klammern. Das Problem der normalen Hintereinanderausführung mit Trennung durch Semikolon ist die Umleitung von Standardeingabe und Standardausgabe, z. B.:

```
pwd > out ; who >> out ; ls >> out
```

Die Umleitung lässt sich auch auf die Fehlerausgabe erweitern:

```
echo "Fehler!"      # geht nach stdout
echo "Fehler!" 1>&2 # geht nach stderr
```

Kommandos lassen sich zur gemeinsamen E/A-Umleitung mit {...} klammern:

{ *Kommando1* ; *Kommando2* ; *Kommando3* ; ... ; }

Wichtig: Die geschweiften Klammern müssen von Leerzeichen eingeschlossen werden!

Die Ausführung der Kommandos erfolgt nacheinander innerhalb der aktuellen Shell, die Ausgabe kann gemeinsam umgelenkt werden, z. B.:

```
{ pwd ; who ; ls ; } > out
```

Die schließende Klammer muss entweder durch ein Semikolon vom letzten Kommando getrennt werden oder am Beginn einer neuen Zeile stehen. Die geschweiften Klammern sind ein ideales Mittel, die Ausgabe aller Programme eines Shell-Skripts gemeinsam umzuleiten, wenn das Skript beispielsweise mit `nohup` oder über `cron` gestartet wird. Man fügt lediglich am Anfang eine Zeile mit der öffnenden Klammer ein und am Schluss die gewünschte Umleitung, z. B.:

```
{
. . .
. . .
. . .
} | mailx -s öutput from Foo" $LOGNAME
```

Eine Folge von Kommandos kann aber auch in einer eigenen Subshell ausgeführt werden:

(*Kommando1* ; *Kommando2* ; *Kommando3* ; ...)

Das Ergebnis des letzten ausgeführten Kommandos wird als Ergebnis der Klammer zurückgegeben. Auch hier kann die Umleitung der Standard-Handles gemeinsam erfolgen. Auch dazu ein Beispiel. Bei unbewachten Terminals lässt sich schön an der Datei `.profile` des Users „basteln“. Eine Zeile

```
( sleep 300 ; audioplay /home/sounds/telefon.au ) &
```

ist schnell eingebaut. Fünf Minuten nach dem Login rennt dann jemand zum Telefon (geht natürlich nur, wenn der Computer auch soundfähig ist). Noch gemeiner wäre

```
( sleep 300 ; kill -9 0 ) &
```

Abschließend vielleicht noch etwas Nützliches: Wenn Sie feststellen, dass eine Plattenpartition zu klein geworden ist, müssen Sie nach Einbau und Formatierung einer neuen Platte oftmals ganze Verzeichnisbäume von der alten Platte auf die neue kopieren. Auch hier hilft die Kommandoverkettung zusammen mit dem Programm *tar* (*Tape ARchive*), das es nicht nur erlaubt, einen kompletten Verzeichnisbaum auf ein Gerät, etwa einen Streamer, zu kopieren, sondern auch in eine Datei oder auf die Standardausgabe. Man verknüpft einfach zwei *tar*-Prozesse, von denen der erste das Verzeichnis archiviert und der zweite über eine Pipe das Archiv wieder auspackt. Der Trick daran ist, dass beide Prozesse in verschiedenen Verzeichnissen arbeiten. Angenommen, Sie wollen das Verzeichnis `/usr/local` nach `/mnt` kopieren:

```
( cd /usr/local ; tar cf - . ) > ( cd /mnt ; tar xvf - )
```

Der Parameter `f` weist *tar* an, auf eine Datei zu schreiben oder von einer Datei zu lesen. Hat die Datei wie oben den Namen `-`, handelt es sich um `stdout` bzw. `stdin`.

4.8.13 Strukturen der Shell

In diesem Abschnitt werden die Programmstrukturen (Bedingungen, Schleifen etc.) behandelt. Zusammen mit den Shell-Variablen und den E/A-Funktionen `echo`, `cat` und `read` hat man nahezu die Funktionalität einer Programmiersprache. Es fehlen lediglich strukturierte Elemente wie z. B. Arrays und Records, die teilweise in anderen Shells (z. B. Korn-Shell) oder auch in Skript-Sprachen realisiert sind.

4.8.14 Bedingungen testen

Das wichtigste Kommando ist `test`, mit dem man mannigfache Bedingungen testen kann.

`test` *Argument* prüft eine Bedingung und liefert `true` (`0`), falls die Bedingung erfüllt ist, und `false` (`1`), falls die Bedingung nicht erfüllt ist. Der Fehlerwert `2` wird nurückgegeben, wenn das Argument syntaktisch falsch ist (meist durch Ersetzung hervorgerufen). Es lassen sich Dateien, Zeichenketten und Integer-Zahlen (16 Bit, bei Linux 32 Bit) überprüfen.

Das *Argument* von `test` besteht aus einer Testoption und einem Operanden, der ein Dateiname oder eine Shell-Variable (Inhalt: String oder Zahl) sein kann. In bestimmten Fällen können auf der rechten Seite eines Vergleichs auch Strings oder Zahlen stehen – bei der Ersetzung von leeren Variablen kann es aber zu Syntaxfehlern kommen. Weiterhin lassen sich mehrere Argumente logisch verknüpfen (UND, ODER, NICHT). Beispiel:

```
test -w /etc/passwd
```

Mit der Kommando-Verkettung lassen sich so schon logische Entscheidungen treffen, z. B:

```
test -w /etc/passwd && echo "Du bist ROOT"
```

Normalerweise kann statt `test` das Argument auch in eckigen Klammern gesetzt werden. Die Klammern müssen von Leerzeichen umschlossen werden:

```
[ -w /etc/passwd ]
```

Die Operationen von Tabelle 4.6 können bei `test` bzw. `[ ... ]` verwendet werden.

4.8.15 Bedingte Anweisung (`if - then - else`)

Wichtig: Als Bedingung kann nicht nur der `test`-Befehl, sondern eine beliebige Folge von Kommandos verwendet werden. Jedes Kommando liefert einen Error-Code zurück, der bei erfolgreicher Ausführung gleich Null (`true`) und bei einem Fehler oder Abbruch ungleich Null (`false`) ist. Zum Testen einer Bedingung dient die `if`-Anweisung. Jede Anweisung muss entweder in einer eigenen Zeile stehen oder durch ein Semikolon von den anderen Anweisungen getrennt werden. Trotzdem verhalten sich bedingte Anweisungen – oder auch die Schleifenkonstrukte, die weiter unten behandelt werden – wie eine einzige Anweisung. Somit ergibt sich eine starke Ähnlichkeit mit der Blockstruktur von *C* oder *Pascal*. Man kann dies ausprobieren, indem man eine `if-` oder `while`-Anweisung interaktiv eingibt. Solange nicht `fi` bzw. `done` eingetippt wurde, erhält man den PS2-Prompt (`>`).

Tabelle 4.6: Bedingungen für `test`

Ausdruck	Bedeutung
`-e` <*datei*>	*datei* existiert
`-r` <*datei*>	*datei* existiert und Leserecht
`-w` <*datei*>	*datei* existiert und Schreibrecht
`-x` <*datei*>	*datei* existiert und Ausführungsrecht
`-f` <*datei*>	*datei* existiert und ist einfache Datei
`-d` <*datei*>	*datei* existiert und ist Verzeichnis
`-h` <*datei*>	*datei* existiert und ist symbolischer Link
`-c` <*datei*>	*datei* existiert und ist zeichenorientiertes Gerät
`-b` <*datei*>	*datei* existiert und ist blockorientiertes Gerät
`-p` <*datei*>	*datei* existiert und ist benannte Pipe
`-u` <*datei*>	*datei* existiert und für Eigentümer s-Bit gesetzt
`-g` <*datei*>	*datei* existiert und für Gruppe s-Bit gesetzt
`-k` <*datei*>	*datei* existiert und t- oder sticky-Bit gesetzt
`-s` <*datei*>	*datei* existiert und ist nicht leer
`-L` <*datei*>	*datei* ist symbolischer Link
`-t` <*d.kennzahl*>	*d.kennzahl* ist einem Terminal zugeordnet
`-n` <*String*>	wahr, wenn *String* nicht leer
`-z` <*String*>	wahr, wenn *String* leer
<*String1*>`=`<*String2*>	wahr, wenn Zeichenketten gleich
<*String1*>`!=`<*String2*>	wahr, wenn Zeichenketten verschieden
`-eq`	(*equal*) gleich
`-ne`	(*not equal*) ungleich
`-ge`	(*greater than or equal*) größer gleich
`-gt`	(*greater than*) größer
`-le`	(*less than or equal*) kleiner gleich
`-lt`	(*less than*) kleiner
UND-Verknüpfung	<*Bedingung1*> `-a` <*Bedingung2*>
ODER-Verknüpfung	<*Bedingung1*> `-o` <*Bedingung2*>
Negation	`!` <*Ausdruck*>
Klammern	`\(` <*Ausdruck*> `\)`

Einseitiges `if`

```
if kommandoliste
then
```

```
    kommandos
fi
```

Zweiseitiges `if`

```
if kommandoliste
then
    kommandos
else
    kommandos
fi
```

Mehrstufiges `if`

```
if kommandoliste1
then
    kommandos
elif kommandoliste2
    then
        kommandos
elif ...
        ...
fi
```

Beispiele für die `if`-Anweisung

Zum Anfang ein „if-Test“ (einige der Kommandos in den Beispielen werden erst später behandelt):

```
echo Bitte eine erste Zahl:
read x1
echo Bitte eine zweite Zahl:
read x2
if [ $x1 -lt $x2 ]
  then
  echo $x1 ist kleiner als $x2
else
  if [ $x1 -eq $x2 ]
    then echo "x1 = x2  \(beide sind gleich $x1\)"
  else echo $x1 ist groesser $x2
  fi
fi
```

Das folgende Skript testet, ob eine als Aufrufparameter angegebene Datei vorhanden ist:

```
echo -n "Die Datei $1 ist "
if [ ! \( -f $1 -o -d $1 \) ]
  then echo -n "nicht "
fi
echo "vorhanden"
```

Es soll eine Meldung ausgegeben werden, falls mehr als fünf Benutzer eingeloggt sind:

```
USERS=`who | wc -l`  # Zeilen der who-Ausgabe zählen
if test $USERS -gt 5
then
  echo "Mehr als 5 Benutzer am Geraet"
fi
```

Das geht natürlich auch kürzer und ohne Backtics:

```
if [ $(who | wc -l) -gt 5 ] ; then
  echo "Mehr als 5 Benutzer am Geraet"
fi
```

Man sollte bei der Entwicklung von Skripten aber ruhig mit der Langfassung beginnen und sich erst der Kurzfassung zuwenden, wenn man mehr Übung hat und die Langfassungen auf Anhieb funktionieren. Ein weiteres Beispiel zeigt eine Fehlerprüfung:

```
if test $# -ne 2
then
  echo "usage: sortiere quelldatei zieldatei"
else
  sort +1 -2 $1  > &2
fi
```

Das nächste Beispiel zeigt eine mehr oder weniger intelligente Anzeige für Dateien und Verzeichnisse. `show` zeigt bei Dateien den Inhalt mit `less` an, während Verzeichnisse mit `ls` präsentiert werden. Fehlt der Parameter, wird interaktiv nachgefragt:

```
if [ $# -eq 0 ]                         # falls keine Angabe
then                                    # interaktiv erfragen
   echo -n "Bitte Namen eingeben: "
   read DATEI
else
   DATEI=$1
fi
if  [-f $DATEI ]                        # wenn normale Datei
then                                    # dann ausgeben
   less $DATEI
elif [ -d $DATEI ]                      # wenn aber Verzeichnis
```

```
  then                                    # dann Dateien zeigen
     ls -CF $DATEI
   else                                   # sonst Fehlermeldung
     echo "cannot show $DATEI"
fi
```

Das nächste Beispiel hängt eine Datei an eine andere Datei an; vorher erfolgt eine Prüfung der Zugriffsberechtigungen. Unser Kommando `append` *Datei1 Datei2*:

```
if [ -r $1 -a -w $2 ]
then
   cat $1 >> $2
else
    echo "cannot append"
fi
```

Beim Vergleich von Zeichenketten sollten möglichst die Anführungszeichen (`''` `...''`) verwendet werden, da sonst bei der Ersetzung durch die Shell unvollständige Test-Kommandos entstehen können. Dazu ein Beispiel:

```
if [ ! -n $1 ] ; then
   echo "Kein Parameter"
fi
```

Ist `$1` wirklich nicht angegeben, wird das Kommando reduziert zu:

```
if [ ! -n ] ; then ....
```

Es ist also unvollständig und es erfolgt eine Fehlermeldung. Dagegen liefert

```
if [ ! -n "$1" ] ; then
    echo "Kein Parameter"
fi
```

bei fehlendem Parameter den korrekten Befehl `if [ ! -n "..."]`. Bei fehlenden Anführungszeichen werden auch führende Leerzeichen der Variablenwerte oder Parameter eliminiert.

Noch ein Beispiel: Es kommt bisweilen vor, dass eine *Userid* wechselt oder dass die Gruppenzugehörigkeit von Dateien geändert werden muss. In solchen Fällen helfen die beiden folgenden Skripten:

```
!/bin/sh
# Change user-id
#
if [ $# -ne 2 ] ; then
  echo "usage `basename $0` <old id> <new id>"
  exit
fi
find / -user $1 -exec chown $2 {} ";"
```

```
#!/bin/sh
# Change group-id
#
if [ $# -ne 2 ] ; then
  echo "usage ‘basename $0‘ <old id> <new id>"
  exit
fi
find / -group $1 -exec chgrp $2 {} ";"
```

4.8.16 Mehrfachauswahl mit `case`

Diese Anweisung wird auch deshalb gerne verwendet, weil sie Muster mit Jokerzeichen und mehrere Muster für eine Auswahl erlaubt.

```
case selector in
   Muster-1) Kommandofolge1 ;;
   Muster-2) Kommandofolge2 ;;
   ...
   Muster-n) Kommandofolgen ;;
esac
```

Die Variable `selector` (*String*) wird der Reihe nach mit den Mustern *Muster-1* bis *Muster-n* verglichen. Bei Gleichheit wird die nachfolgende Kommandofolge ausgeführt und dann nach der `case`-Anweisung (also hinter dem `esac`) fortgefahren.

- In den Mustern sind Metazeichen (`*`, `?`, `[]`) erlaubt, im Selektor dagegen nicht.
- Das Muster `*` deckt sich mit jedem Selektor (Default-Ausgang) und muss als letztes Muster in der `case`-Konstruktion stehen.
- Vor der Klammer können mehrere Muster, getrennt durch `|`, stehen. Das Zeichen `|` bildet eine Oder-Bedingung:

  ```
  case selector in
     Muster-1) Kommandofolge1 ;;
     Muster-2 | Muster3) Kommandofolge2 ;;
     *) Kommandofolge3 ;;
  esac
  ```

Beispiel: Menü mit interaktiver Eingabe:

```
while :  # Endlosschleife (s. später)
do
tput clear  # Schirm löschen und Menütext ausgeben
```

```
   echo " +--------------------------------+"
   echo " | 0 --> Ende                      |"
   echo " | 1 --> Datum und Uhrzeit         |"
   echo " | 2 --> aktuelles Verzeichnis     |"
   echo " | 3 --> Inhaltsverzeichnis        |"
   echo " | 4 --> Mail                      |"
   echo "+---------------------------------+"
   echo "Eingabe: \c"  # kein Zeilenvorschub
   read ANTW
   case $ANTW in
    0) kill -9 0 ;; # und tschuess
    1) date ;;
    2) pwd ;;
    3) ls -CF ;;
    4) elm ;;
    *) echo "Falsche Eingabe!" ;;
   esac
done
```

4.8.17 Die `for`-Anweisung

Diese Schleifenanweisung hat zwei Ausprägungen, mit einer Liste der zu bearbeitenden Elemente oder mit den Kommandozeilen-Parametern.

Die `for`-Schleife mit Liste

```
for selector in liste
   do
   Kommandofolge
done
```

Die Selektor-Variable wird nacheinander durch die Elemente der Liste ersetzt und die Schleife mit der Selektor-Variablen ausgeführt. Beispiele:

```
for X in hans heinz karl luise # vier Listenelemente
do
  echo $X
done
```

Das Programm hat folgende Ausgabe:

```
hans
heinz
karl
luise
```

```
for FILE in *.txt # drucke alle Textdateien
  do              # im aktuellen Verzeichnis
  lpr $FILE
done


for XX in $VAR    # geht auch mit
  do
  echo $XX
done
```

Die `for`-Schleife mit Kommandozeilen-Parametern

for selector
   do
   Kommandofolge
done

Die Selektor-Variable wird nacheinander durch die Parameter `$1` bis `$n` ersetzt und mit diesen Werten die Schleife durchlaufen. Es gibt also `$#` Schleifendurchläufe. Beispiel:

Das Skript `makebak` erzeugt für die in der Parameterliste angegebenen Dateien eine `.bak`-Datei:

```
for FF
  do
  cp $FF ${FF}.bak
done
```

4.8.18 Abweisende Wiederholungsanweisung (`while`)

Als Bedingung kann nicht nur eine „klassische" Bedingung (`test` oder `[]`), sondern selbstverständlich auch der Ergebniswert eines Kommandos oder einer Kommandofolge verwendet werden.

while *Bedingung*
   do
   Kommandofolge
done

Solange der Bedingungsausdruck den Wert `true` liefert, wird die Schleife ausgeführt. Beispiele:

Warten auf eine Datei (z. B. vom Hintergrundprozess):

```
while [ ! -f foo ]
  do
  sleep 10  # Wichtig damit die Prozesslast nicht zu hoch wird
done
```

Pausenfüller für das Terminal – Abbruch mit (Strg-C):

```
while :
  do
  tput clear  # BS löschen
  echo "\n\n\n\n\n" # 5 Leerzeilen
  banner $(date '+ \%T ') # Uhrzeit groß
  sleep 1  # 1s Pause
done
```

Umbenennen von Dateien durch Anhängen eines Suffixes:

```
# Aufruf change suffix datei(en)
if [ $# -lt 2 ] ; then
   echo "usage: `basename $0` suffix file(s)"
else
   SUFF=$1                # Suffix speichern
   shift
   while [ $# -ne 0 ]     # solange Parameter da sind
     do
     mv $1 ${1}.$SUFF     # umbenennen
     shift
   done
fi
```

Umbenennen von Dateien durch Anhängen eines Suffixes – Variante 2 mit `for`:

```
 Aufruf change suffix datei(en)
if [ $# -lt 2 ] ; then
   echo "usage: `basename $0` suffix file(s)"
else
   SUFF=$1                      # Suffix speichern
   shift
   for FILE
     do
     mv $FILE ${FILE}.$SUFF     # umbenennen
     shift
   done
fi
```

4.8.19 `until`-Anweisung

Diese Anweisung ist identisch zu einer `while`-Schleife mit negierter Bedingung. Als Bedingung kann nicht nur eine „klassische“ Bedingung (`test` oder `[]`),

sondern selbstverständlich auch der Ergebniswert eines Kommandos oder einer Kommandofolge verwendet werden:

`until` *Bedingung*
  `do`
  Kommandofolge
`done`

Die Schleife wird solange abgearbeitet, bis *Bedingung* einen Wert ungleich Null liefert. Beispiele:

```
# warten, bis sich der Benutzer hans eingeloggt hat
until [ ‘who | grep -c "hans" -gt 0 ]
  do
  sleep 10
done
```

4.8.20 Weitere Anweisungen

- `exit`
  Wie die interaktive Shell kann auch ein Shell-Skript mit `exit` abgebrochen werden. Vom Terminal aus kann mit der Taste (Strg-C) abgebrochen werden, sofern das Signal nicht abgefangen wird (siehe `trap`).
- `break [n]`
  Verlassen von `n` umfassenden Schleifen; Voreinstellung für `n` ist `1`.
- `continue [n]`
  Beginn des nächsten Durchgangs der `n`-ten umfassenden Schleife, d. h. der Rest der Schleife(n) wird nicht mehr ausgeführt; Voreinstellung für `n` ist `1`.
- Interne Kommandos
  Etliche der besprochenen Shell-Kommandos starten nicht, wie sonst üblich, einen eigenen Prozess, sondern sie werden direkt von der Shell interpretiert und ausgeführt. Teilweise ist keine E/A-Umleitung möglich. Einige Kommandos der folgenden Auswahl wurden bereits besprochen, andere werden weiter unten behandelt. Zum Teil gibt es interne und externe Versionen, z. B. `echo` (intern) und `/bin/echo` (extern).
  - `break` Schleife verlassen
  - `continue` Sprung zum Schleifenanfang
  - `echo` Ausgabe
  - `eval` Mehrstufige Ersetzung
  - `exec` Überlagerung der Shell durch ein Kommando
  - `exit` Shell beenden

  - `export` Variablen für Subshells bekannt machen
  - `read` Einlesen einer Variablen
  - `shift` Parameterliste verschieben
  - `trap` Behandlung von Signalen

- `set` [*Optionen*] [*Parameterliste*]
  Setzen von Shell-Optionen und Positionsparametern (`$1 ...$n`). Einige Optionen:

  `v` gibt die eingelesenen Shell-Eingaben auf dem Bildschirm aus.

  `x` gibt alle Kommandos vor der Ausführung aus (zeigt Ersetzungen).

  `n` liest die Kommandos von Shell-Skripten, führt sie jedoch nicht aus.

  Der Aufruf von `set` ohne Parameter liefert die aktuelle Belegung der Shell-Variablen. Außerdem kann `set` verwendet werden, um die Positionsparameter zu besetzen.

  `set eins zwei drei vier` besetzt die Parameter mit `$1=eins, $2=zwei, $3=drei` und `$4=vier`. Da dabei auch Leerzeichen, Tabs, Zeilenwechsel und anderes „ausgefiltert“ wird (genauer alles, was in der Variablen `IFS` steht), ist `set` manchmal einfacher zu verwenden als die Zerlegung einer Zeile mit `cut`. Die Belegung der Parameter kann auch aus einer Variablen (z. B. `set $VAR`) oder aus dem Ergebnis eines Kommando-Aufrufs erfolgen. Beispiel:

```
set ‘date‘           # $1=Thu $2=Nov $3=10 $4=10:44:16 $5=MEZ $6=2005
echo "Es ist $4 Uhr"
Es ist 10:44:16 Uhr
```

  Aber es gibt Fallstricke: Wenn man beispielsweise den Output von `ls` bearbeiten möchte, gibt es zunächst unerklärliche Fehlermeldungen (`set: unknown option`):

```
ls -l > foo
echo "Dateiname Laenge"
while read LINE
  do
  set $LINE
  echo $9 $5
done < foo
rm foo
```

  Da die Zeile mit dem Dateityp und den Zugriffsrechten beginnt und für normale Dateien ein „`-`“ am Zeilenbeginn steht, erkennt `set` eine falsche Option (z. B. `-rwxr-xr-x`). Abhilfe schafft das Voranstellen eines Buchstabens:

```
ls -l > foo
echo "Dateiname Laenge"
while read LINE
  do
```

```
  set Z$LINE
  echo $9 $5
done < foo
rm foo
```

Weitere Beispiele: Wenn ein Benutzer eingeloggt ist, wird ausgegeben, seit wann. Sonst erfolgt eine Fehlermeldung.

```
if HELP=‘who | grep $1‘
then
  echo -n "$1 ist seit "
  set $HELP
  echo "$5 Uhr eingeloggt."
else
  echo "$1 ist nicht auffindbar"
fi
```

Ersetzen der englischen Tagesbezeichung durch die deutsche:

```
set ‘date‘
case $1 in
  Tue) tag=Die;;
  Wed) tag=Mit;;
  Thu) tag=Don;;
  Sat) tag=Sam;;
  Sun) tag=Son;;
  *)   tag=$1;;
esac
echo $tag $3.$2 $4 $6 $5
```

Arithmetik in Skripten

Die `expr`-Anweisung erlaubt das Auswerten von arithmetischen Ausdrücken. Das Ergebnis wird in die Standardausgabe geschrieben. Als Zahlen können 16-Bit-Integerzahlen (beim Ur-Unix) oder 32-Bit-Integerzahlen (bei Linux) verwendet werden (bei manchen Systemen auch noch längere Zahlen mit 64 Bit).

Die *Bash* wurde auch an dieser Stelle erweitert. Mit der doppelten Klammerung `$((` *Ausdruck* `))` kann man rechnen, ohne ein externes Programm aufzurufen. `expr` *Ausdruck* und `$((` *Ausdruck* `))` beherrschen die vier Grundrechenarten:

Arithmetische Operationen	
+	Addition
-	Subtraktion
*	Multiplikation
/	Division
%	Divisionsrest (Modulo-Operator)

Die Priorität „Punkt vor Strich“ gilt auch hier. Außerdem können Klammern gesetzt werden. Da die Klammern und der Stern auch von der Shell verwendet

werden, müssen diese Operationszeichen immer durch den Backslash geschützt werden: `\* , \( ...\)`

Damit die Operatoren von der Shell erkannt werden, müssen sie von Leerzeichen eingeschlossen werden, zum Beispiel eine Zuweisung der Summe von `A` und `B` an `X` durch:

```
X=‘expr $A + $B‘
```

oder

```
X=$((expr $A + $B))
```

(Backquotes beachten!) Außerdem sind logische Operationen implementiert, die den Wert `0` für `wahr` und den Wert `1` für `falsch` liefern:

Vergleichsoperationen	
expr1 `|` *expr2*	oder
expr1 `&` *expr2*	und
expr1 `<` *expr2*	kleiner
expr1 `<=` *expr2*	kleiner oder gleich
expr1 `>` *expr2*	größer
expr1 `>=` *expr2*	größer oder gleich
expr1 `=` *expr2*	gleich
expr1 `!=` *expr2*	unleich

Beispiel: Nimm-Spiel, interaktiv:

```
ANZ=0
if test $# -ne 1
then
  echo "usage: $0  Startzahl"
else
  echo "NIM-Spiel als Shell-Skript"
  echo "Jeder Spieler nimmt abwechselnd 1, 2 oder 3 Hoelzer"
  echo "von einem Haufen, dessen Anfangszahl beim Aufruf festgelegt"
  echo "wird. Wer das letzte Holz nimmt, hat verloren."
  echo
  ANZ=$1
  while [ $ANZ -gt 1 ]                # bis nur noch 1 Holz
    do                                # da ist wiederholen
    echo "\nNoch $ANZ Stueck. Du nimmst (1 - 3): \c # Benutzer
    read N
    if [ $N -lt 1 -o $N -gt 3 ] ; then # Strafe bei Fehleingabe
      N=1
    fi
    ANZ=‘expr $ANZ - $N‘              # Benutzer nimmt N weg
    if [ $ANZ -eq 1 ] ; then          # Computer muß letztes Holz nehmen
      echo "\nGratuliere, Du hast gewonnen"
```

```
      exit                            # Prozedur verlassen
    else
      C=`expr \( $ANZ + 3 \) \% 4     # Computerzug berechnen
      if [ $C -eq 0 ] ; then
        C=1                           # Wenn 0 Verlustposition
      fi
      echo "Es bleiben $ANZ Stueck. Ich nehme ${C}.\c"
      ANZ=`expr $ANZ - $C`            # Computerzug abziehen
      echo " Rest $ANZ"
    fi
  done                                # Dem Benutzer bleibt
  echo "\nIch habe gewonnen"          # das letzte Holz
fi
```

4.8.21 `exec` [*Kommandozeile*]

Ähnlich wie beim Dot-Kommando wird keine Subshell erzeugt, sondern die Kommandozeile in der aktuellen Umgebung ausgeführt. Eine erste Anwendung liegt darin, das aktuelle Programm durch ein anderes zu überlagern. Wenn Sie z. B. die *Bourne-Shell* als Login-Shell haben, aber lieber mit der *Bash* arbeiten, können Sie die *Bourne-Shell* durch die Kommandozeile

```
exec /bin/bash
```

als letzte Zeile in der `.profile`-Datei durch die *Bash* ersetzen. (Wenn Sie die *Bash* nur aufrufen würden, müßten Sie beide Shells beenden, um sich auszuloggen.) Das Kommando entspricht also dem Systemcall `exec()`. Wird jedoch kein Kommando angegeben, kann die E/A der aktuellen Shell dauerhaft umgeleitet werden. Beispiel:

```
exec 2>fehler
```

leitet alle folgenden Fehlerausgaben in die Datei `fehler` um, bis die Umleitung explizit durch

```
exec 2>-
```

zurückgenommen wird. Es können bei `exec` auch andere Dateideskriptoren verwendet werden. Ebenso kann auch die Dateiumleitung einer Eingabedatei erfolgen, z. B.:

```
exec 3< datei
```

Danach kann mit `read <&3` von dieser Datei gelesen werden, bis die Umleitung mit `exec 3<-` wieder zurückgenommen wird. Man kann also in Shell-Skripten durch das Einfügen einer `exec`-Anweisung die Standardausgabe/-eingabe global für das gesamte Skript umleiten, ohne weitere Änderungen vornehmen zu

müssen (eine andere Möglichkeit wäre die oben beschriebene Verwendung von { }).

4.8.22 `eval` [*Argumente*]

Das Kommando `eval` liest seine Argumente, wobei die üblichen Ersetzungen stattfinden, und führt die resultierende Zeichenkette als Kommando aus. Die Argumente der Kommandozeile werden von der Shell gelesen, wobei Variablen- und Kommando- sowie Dateinamenersetzungen durchgeführt werden. Die sich ergebende Zeichenkette wird anschließend erneut von der Shell gelesen, wobei wiederum die oben genannten Ersetzungen durchgeführt werden. Schließlich wird das resultierende Kommando ausgeführt. Beispiel:

```
user$ A='Hello world!'
user$ X='$A'
user$ echo $X
$A
user$ eval echo $X
Hello world!
```

Der Rückgabestatus von `eval` ist der Rückgabestatus des ausgeführten Kommandos oder `0`, wenn keine Argumente angegeben wurden. Ein weiteres Beispiel:

```
user$ cat /etc/passwd | wc -l
76
user$ foo='cat /etc/passwd'
user$ bar=' | wc -l'
user$ $foo $bar
cat: | : No such file or directory
cat: wc: No such file or directory
cat: -l: No such file or directory
user$ eval $foo $bar
76
```

In diesem Beispiel wird zunächst ein einfaches Kommando gestartet, das die Anzahl der Zeilen der Datei `/etc/passwd` bestimmt. Anschließend werden die beiden Teile des gesamten Kommandos in die zwei Shell-Variablen `foo` und `bar` aufgeteilt. Der erste Aufrufversuch `$foo $bar` bringt nicht das gewünschte Ergebnis, sondern lediglich einige Fehlermeldungen, da in diesem Fall der Wert von `bar` als Argument für `foo` interpretiert wird (`cat` wird mit den Dateien `/etc/passwd`, `|`, `wc` und `-l` aufgerufen). Wird jedoch das Kommando `eval` auf die Argumente `$foo` und `$bar` angewendet, werden diese zunächst zur Zeichenkette `cat /etc/passwd | wc -l` ersetzt. Diese Zeichenkette wird dann durch das Kommando `eval` erneut von der Shell gelesen, die jetzt das Zeichen `|` in der Kommandozeile als Pipe-Symbol erkennt und das Kommando ausführt. Das

Kommando `eval` wird üblicherweise dazu eingesetzt, eine Zeichenkette als Kommando zu interpretieren, wobei zweifach Ersetzungen in den Argumenten der Kommandozeile vorgenommen werden.

Eine andere Anwendung ist beispielsweise die Auswahl des letzten Parameters der Kommandozeile. Mit `\$$#` erhält man die Parameterangabe (bei fünf Parametern `$5`). Das erste Dollarzeichen wird von der Shell ignoriert (wegen des Backslash), `$#` hingegen ausgewertet. Durch `eval` wird der Ausdruck nochmals ausgewertet, man erhält so den Wert des letzten Parameters:

```
eval \$$#
```

Aber Vorsicht, das funktioniert nur bei 1-9 Parametern, denn z. B. der zwölfte Parameter führt zu `$12` → `${1}2`. Es lassen sich mit `eval` sogar Pointer realisieren. Falls die Variable `PTR` den Namen einer anderen Variablen, z. B. `XYZ`, enthält, kann auf den Wert von `XYZ` durch `eval $PTR` zurückgegriffen werden, z. B. durch

```
eval echo \$$PTR
```

4.8.23 `trap` *´Kommandoliste´ Signale*

Ausführen der Kommandoliste, wenn eines der angegebenen Signale an den Prozess (= Shell) gesendet wird. Die Signale werden in Form der Signalnummern oder über ihre Namen (`SIGKILL`, `SIGHUP` etc.), getrennt durch Leerzeichen, aufgeführt.

Ist die Kommandoliste leer, werden die entsprechenden Signale abgeschaltet. Bei einfachen Kommandos reichen oft auch die Anführungszeichen, um die Shell-Ersetzung zu verhindern.

Signale sind eine Möglichkeit, über die verschiedenen Prozesse, also laufende Programme, miteinander kommunizieren können. Ein Prozess kann einem anderen Prozess ein Signal senden (der Betriebssystemkern spielt dabei den „Postboten"). Der Empfängerprozess reagiert auf das Signal, z. B. dadurch, dass er sich beendet. Der Prozess kann das Signal auch ignorieren. Das ist beispielsweise nützlich, wenn ein Shell-Skript nicht durch den Benutzer von der Tastatur aus abgebrochen werden soll. Mit dem `trap`-Kommando kann man festlegen, mit welchen Kommandos auf ein Signal reagiert werden soll bzw. ob überhaupt reagiert werden soll. Die Datei `/usr/include/Signal.h` enthält eine Liste aller Signale. Neben anderen können folgende Signalnummern verwendet werden:

Typische Signale

0 (SIGKILL)	Kill: Beenden der Shell
1 (SIGHUP)	Hangup: Beenden der Verbindung zum Terminal/Modem
2 (SIGINT)	Interrupt: Wie Strg-C am Terminal
3 (SIGQUIT)	Quit: Beenden von der Tastatur aus
9 (SIGKILL)	Kill: Kann nicht abgefangen werden; beendet immer den empfangenden Prozess
15 (SIGTERM)	Terminate: Software-Terminate, Voreinstellung

Beispiele:

```
 Skript sperren gegen Benutzerunterbrechung:
trap "" 2 3
```

oder auch

```
# Skript sauber beenden
trap 'rm tmpfile; cp foo xyz; exit' 0 2 3 15
```

Bitte nicht das `exit`-Kommando am Schluss vergessen, sonst wird das Skript nicht beendet. Wiedereinschalten der Signale erfolgt durch `trap [`*Signale*`]`. Ein letztes Beispiel zu `trap`:

```
# Automatisches Ausführen des Shell-Skripts .logoff beim
# Ausloggen durch den folgenden Eintrag in .profile:
trap .logoff 0
```

4.8.24 Shell-Funktionen

Shell-Funktionen bieten eine weitere Strukturierungsmöglichkeit. Funktionen können in Shell-Skripten, aber auch interaktiv definiert werden. Sie lassen sich jedoch nicht wie Variablen exportieren, gelten also nur in der aktuellen Shell. Sie werden nach folgender Syntax definiert:

Funktionsname `()`
  {
  Kommandofolge
  }

Steht die schließende geschweifte Klammer nicht in einer eigenen Zeile, gehört ein Semikolon davor. Die runden Klammern hinter dem Funktionsnamen teilen dem Kommandozeilen-Interpreter der Shell mit, dass nun eine Funktion definiert werden soll (und nicht ein Kommando *Funktionsname* aufgerufen wird). Es kann keine Parameterliste in den Klammern definiert werden.

Der Aufruf der Shell-Funktion erfolgt durch Angabe des Funktionsnamens, gefolgt von Parametern (genauso wie der Aufruf eines Skripts). Die Parameter werden innerhalb der Funktion genauso wie beim Aufruf von Shell-Skripten

über `$1` bis `$n` angesprochen. Ein Wert kann mit der Anweisung `return` <*Wert*> zurückgegeben werden, er ist über den Parameter `$?` abfragbar. Beispiel:

```
isdir () # testet, ob $1 ein Verzeichnis ist
   {
   if [ -d $1 ] ; then
     echo "$1 ist ein Verzeichnis" # Kontrolle zum Test
     return 0
   else
     return 1
    fi
   }
```

Im Gegensatz zum Aufruf von Shell-Skripten werden Funktionen *in der aktuellen Shell* ausgeführt und sie können bei der *Bourne-Shell* nicht exportiert werden; die *Bash* erlaubt dagegen das Exportieren mit `export -f`. Das folgende Beispiel illustriert die Eigenschaften von Shell-Funktionen.

Die folgende Funktion gibt den Eingangsparameter in römischen Zahlen aus. Dabei wird die Zahl Schritt für Schritt in der Variablen `ZAHL` zusammengesetzt. Würde man bei der Funktion `ZIFF` ein Skript verwenden, ginge das nicht, da sich der Wert von `ZAHL` ja nicht aus dem aufgerufenen Skript heraustransportieren ließe.

```
#
# Ausgabe des Eingangsparameters $1 in roemischen Ziffern
#
ZIFF ()

   # Funktion zur Bearbeitung einer einstelligen Ziffer $1
   # Einer-, Zehner-, Hunderterstelle unterscheiden sich nur
   # durch die verw. Zeichen $2: Einer, $3: Fuenfer, $4: Zehner
   { X=$1
   if test $X -eq 9; then
      ZAHL=${ZAHL}$2$4
   elif test $X -gt 4; then
      ZAHL=${ZAHL}$3
      while test $X -ge 6; do
         ZAHL=${ZAHL}$2 ; X=`expr $X - 1`
      done
   elif test $X -eq 4; then
    ZAHL=${ZAHL}$2$3
  else
      while test $X -gt 0; do
          ZAHL=${ZAHL}$2 ; X=`expr $X - 1`
      done
    fi
    }
if test $# -eq 0; then
      echo "usage: roem Zahl"; exit
```

```
fi
XX=$1
while test $XX -gt 999; do
      ZAHL=${ZAHL}"M"; XX=`expr $XX - 1000`
done
ZIFF `expr $XX / 100` C D M
XX=`expr $XX \% 100`
ZIFF `expr $XX / 10` X L C
ZIFF `expr $XX \% 10` I V X
echo "$ZAHL \n"
```

4.8.25 xargs

Das `xargs`-Programm ist ein eigenständiges Kommando, das hier behandelt wird, weil es – ebenso wie das unten behandelte `find`-Kommando sehr häufig in Shell-Skripten verwendet wird. `xargs` übergibt alle aus der Standardeingabe gelesenen Daten einem Programm als zusätzliche Argumente. Der Programmaufruf wird einfach als Parameter von `xargs` angegeben:

`xargs` *Programm* [*Parameter*]

Ein Beispiel soll die Funktionsweise klar machen:

```
user$ ls *.txt | xargs echo Textdateien
```

Hier erzeugt `ls` eine Liste aller Dateien mit der Endung `.txt` im aktuellen Verzeichnis. Das Ergebnis wird über die Pipe an `xargs` weitergereicht. `xargs` ruft `echo` mit den Dateinamen von `ls` als zusätzliche Parameter auf. Der Output ist dann

```
Textdateien: kap1.txt kap2.txt kap3.txt h.txt
```

Durch Optionen ist es möglich, die Art der Umwandlung der Eingabe in Argumente durch `xargs` zu beeinflussen. Mit der Option `-n` <*Nummer*> wird eingestellt, mit wievielen Parametern das angegebene Programm aufgerufen werden soll. Fehlt der Parameter, nimmt `xargs` die maximal mögliche Zahl von Parametern. Je nach Anzahl der Parameter ruft `xargs` das angegebene Programm einmal oder mehrfach auf. Dazu ein Beispiel: Vergleich einer Reihe von Dateien nacheinander mit einer vorgegebenen Datei:

```
user$  ls *.dat | xargs -n1 cmp compare Muster
```

Die Vergleichsdatei `Muster` wird der Reihe nach mittels `cmp` mit allen Dateien verglichen, die auf `.dat` enden. Die Option `-n1` veranlasst `xargs`, je Aufruf immer nur einen Dateinamen als zusätzliches Argument bei `cmp` anzufügen.

Mit der Option `- i` <*Zeichen*> ist es möglich, an einer beliebigen Stelle im Programmaufruf, auch mehrfach, anzugeben, wo die eingelesenen Argumente einzusetzen sind. In diesem Modus liest `xargs` jeweils ein Argument aus der

Standardeingabe, ersetzt im Programmaufruf jedes Vorkommen des hinter `-i` angegebenen Zeichens durch dieses Argument und startet das Programm. In dem folgenden Beispiel wird dies benutzt, um alle Dateien mit der Endung `.txt` in `.bak` umzubenennen.

```
user$ ls *.txt | cut -d. f1 | xargs -iP mv P.txt P.bak
```

Das Ganze funktioniert allerdings nur, wenn die Dateien nicht noch weitere Punkte im Dateinamen haben.

`xargs` ist überall dann notwendig, wenn die Zahl der Argumente recht groß werden kann. Obwohl Linux extrem lange Kommandozeilen zulässt, ist die Länge doch begrenzt. `xargs` nimmt immer so viele Daten aus dem Eingabestrom, wie in eine Kommandozeile passen, und führt dann das gewünschte Kommando mit diesen Parametern aus. Liegen weitere Daten vor, wird das Kommando entsprechend oft aufgerufen. Insbesondere mit dem folgenden Kommando sollte `xargs` verwendet werden, da `find` immer den gesamten Dateipfad liefert, also schnell recht lange Argumente weitergibt.

4.8.26 find

Durchsuchen der Platte nach bestimmten Dateien. Dieses Kommando ist sehr mächtig und besitzt zahlreiche Optionen, welche die Bedingungen für die Dateiauswahl (= Treffer) festlegen. Die angegebenen Pfade werden rekursiv durchsucht, d. h. auch Unterverzeichnisse und Unter-Unterverzeichnisse usw. Aufrufschema:

`find` *Pfadname(n) Bedingung(en) Aktion(en)*

Bei den nachfolgenden Optionen (= Bedingungen) steht das `N` für eine Zahlenangabe (ganze Zahl). Das Vorzeichen dieser Zahl bestimmt die Bedingung:

`N` genau `N`

`+N` mehr als `N`

`-N` weniger als `N`

Bedingungen (es werden nur die wichtigsten Optionen aufgeführt):

- `-name` *Dateiname*
  Suche nach bestimmten Dateien (bei Verwendung von Metazeichen wie `*` oder `?` den Namen unbedingt in `'...'` einschließen).
- `-type` *T*
  Suche nach einem bestimmten Dateityp *T*:

  `f` normale Datei

`d` Directory

`b` Block Device

`c` Character Device

`p` Named Pipe

- `-perm` *onum*
  Suche nach Dateien mit den durch die Oktalzahl *onum* angegebenen Zugriffsrechten. Steht das Zeichen `-` vor *onum*, werden nicht alle, sondern nur die spezifizierten Rechte geprüft (z. B. `Test SUID: -perm -4000`).
- `-links` *N*
  Suche nach Dateien mit einer bestimmten Anzahl *N* von Links
- `-user` *Kennung*
  Suche nach Dateien eines bestimmten Users; es kann der Login-Name oder die UID angegeben werden.
- `-group` *Kennung*
  Wie `-user`, jedoch für Gruppen
- `-size` *N*
  Suche nach Dateien mit *N* Blöcken
- `-mount`
  Suche nur auf dem aktuellen Datenträger (wichtig, falls weitere Platten über NFS eingebunden sind)
- `-newer` *Dateiname*
  Suche nach Dateien, die jünger sind als die angegebene

Aktionen (auch hier nur die wichtigsten):

- `-print`
  Ausgabe der gefundenen Dateinamen auf die Standardausgabe
- `-exec` *Kommando*
  Ausführen eines Kommandos. Wird innerhalb des Kommandos die leere geschweifte Klammer {} aufgeführt, so wird anstelle der geschweiften Klammern der jeweils gefundene absolute Pfad der Datei eingesetzt. Das per `-exec` aufgerufene Kommando wird immer mit einem geschützten Semikolon (\; oder `'';''` oder `';'`) abgeschlossen.
- `-ok` *Kommando*
  Wie `-exec`, jedoch mit Sicherheitsabfrage

Beispiele:

`user$` **find / -name '*.c' -print**

Beginnt, beim Wurzelverzeichnis, alle *C*-Quellen (`.c`) zu suchen und gibt die Namen (und Pfade) auf dem Bildschirm aus. Die Apostrophe verhindern die Ersetzung der Angabe `*.c` durch die Shell.

`user$` **find -user markus -print**

Sucht alle Dateien von User markus

`user$` **find . -name dat1 -print**

Sucht nach allen Vorkommen der Datei `dat1` ab dem aktuellen Verzeichnis

`user$` **find . -print -name dat1**

Gibt *alle* Dateinamen ab dem aktuellen Verzeichnis aus. *Vorsicht Falle*: Da die Bedingungen von links nach rechts ausgewertet werden und `-print` immer `wahr` ist, hat der Teil `-name dat1` keine Wirkung.

`user$` **find . -name '*.bak' -exec rm \;**

Sucht alle `.bak`-Dateien ab dem aktuellen Verzeichnis und löscht sie

`user$` **find /usr -size +2000 -print**

Gibt alle Dateien der Benutzer mit mehr als 2000 Blöcken aus (z. B. um die Platzverschwender zu mahnen)

`user$` **find . -name '*.bak' -ok rm \;**

Sucht alle `.bak`-Dateien ab dem aktuellen Verzeichnis und löscht sie nur, wenn die Nachfrage mit `y` beantwortet wurde.

`user$` **find / -user markus -exec rm \;**

Löscht alle Dateien von User markus, wo auch immer sie stehen. Verwendet man `xargs`, sieht das Kommando so aus:

`user$` **find . -user markus -print | xargs rm**

4.9 Beispiele für Shell-Skripten

Die folgenden Beispiele sollen Ihnen den Weg in die Shell-Programmierung ebnen. Es sind meist relativ kurze Programme, die bestimmte Aspekte illustrieren.

4.9.1 Datei verlängern

...weil immer wieder danach gefragt wird: „Wie hänge ich den Inhalt einer Variablen an eine Datei an?“

```
( cat file1 ; echo "$SHELLVAR" ) > file2
```

4.9.2 Telefonbuch

Man braucht nicht viel Programm, um eine Telefonliste zu verwalten. Sucht man nach einer Nummer, wird das Skript mit einem Namen oder einer Namensliste als Parameter aufgerufen. Damit die Telefonliste nicht verloren gehen kann, hängt sie gleich als Hier-Dokument am Skript dran. Nebenbei – selbst bei etlichen hundert Einträgen ist `grep` immer noch schnell genug:

```
if [ $# -eq 0 ]
then
   echo "usage: ‘basename $0‘ Name [Name ..]"
   exit 2
fi
for SUCH in $*
do
    if [ ! -z $SUCH ] ; then
       grep $SUCH << "EOT"
       Hans  123456
       Fritz 234561
       Karl  345612
       Egon  456123
EOT
   fi
done
```

4.9.3 Argumente mit J/N-Abfrage ausführen

Das folgende Skript führt alle Argumente nach vorheriger Abfrage aus. Mit `j` wird die Ausführung bestätigt, mit `q` das Skript abgebrochen und mit jedem anderen Buchstaben (in der Regel `n`) ohne Ausführung zum nächsten Argument übergegangen. Ein- und Ausgabe erfolgen immer über das Terminal(-fenster), weil `/dev/tty` angesprochen wird. Das Skript wird anstelle der Argumentenliste bei einem anderen Kommando eingesetzt, z. B. Löschen mit Nachfrage durch `rm $(pick *)`

```
# pick - Argumente mit Abfrage liefern
for I ; do
   echo "$I (j/n)? \c" > /dev/tty
   read ANTWORT
```

```
   case $ANTWORT in
     j*|J*) echo $I ;;
     q*|Q*) break ;;
   esac
done </dev/tty
```

4.9.4 Dateien im Pfad suchen

Das folgende Skript bildet das Kommando *which* nach. Es sucht im aktuellen Pfad (durch `PATH` spezifiziert) nach der angegebenen Datei und gibt die Fundstelle aus. An diesem Skript kann man auch eine Sicherheitsmaßnahme sehen. Für den Programmaufruf wird der Pfad neu gesetzt, damit nur auf Programme aus `/bin` und `/usr/bin` zugegriffen wird. Bei Skripten, die vom Systemverwalter für die Allgemeinheit erstellt werden, sollte man entweder so verfahren oder alle Programme mit absolutem Pfad aufrufen.

```
#!/bin/sh
# Suchen im Pfad nach einer Kommando-Datei
OPATH=$PATH
PATH=/bin:/usr/bin
if [ $# -eq 0 ] ; then
   echo "usage: which kommando" ; exit 1
fi
for FILE
do
   for I in ‘echo $OPATH | sed -e ’s/^:/.:/’ -e ’s/::/:.:/g \ -e ’s/:$/:./’‘
   do
      if [ -f "$I/$FILE" ] ; then
        ls -ld "$I/$FILE"
      fi
   done
done
```

4.9.5 Berechnung des Osterdatums nach C. F. Gauss

Eigentlich braucht sowas niemand als Shell-Skript. Das Beispiel soll lediglich zeigen, dass man auch komplexere Ganzzahl-Rechnungen mit der Shell erledigen kann.

```
if [ $# -eq 0 ] ; then
    echo ’Osterdatum fuer Jahr: \c’; read JAHR
else
   JAHR="$1"
fi
G=‘expr $JAHR \% 19 + 1‘
C=‘expr $JAHR / 100 + 1‘
```

```
X=‘expr \( $C / 4 - 4 \) \* 3‘
Z=‘expr \( $C \* 8 + 5 \) / 25 - 5‘
D=‘expr $JAHR \* 5 / 4 - $X - 10‘
E=‘expr \( $G \* 11 + $Z - $X + 20 \) \% 30‘
if test $E -lt 0; then
   $E=‘expr $E + 30‘
fi
if [ $E -eq 25 -a $G -gt 11 -o $E -eq 24 ] ; then
   E=‘expr $E + 1‘
fi
TAG=‘expr 44 - $E‘
if [ $TAG -lt 21 ] ; then
   TAG=‘expr $TAG + 30‘
fi
TAG=‘expr $TAG + 7 - \( $D + $TAG \) \% 7‘
if [ $TAG -gt 31 ] ; then
   TAG=‘expr $TAG - 31‘
   MON=4
else
   MON=3
fi
echo " Ostern $JAHR ist am ${TAG}.${MON}.\n"
```

Statt des `expr`-Befehls kann bei der *Bash* auch das Konstrukt `$(( ...))` verwendet werden. Das Programm sieht dann so aus:

```
if [ $# -eq 0 ] ; then
    echo österdatum fuer Jahr: \c"; read JAHR
else
   JAHR="$1"
fi
G=$(($JAHR \% 19 + 1))
C=$(($JAHR / 100 + 1))
X=$((\( $C / 4 - 4 \) \* 3))
Z=$((\( $C \* 8 + 5 \) / 25 - 5))
D=$(($JAHR \* 5 / 4 - $X - 10))
E=$((\( $G \* 11 + $Z - $X + 20 \) \% 30))
if test $E -lt 0; then
   $E=$(($E + 30))
fi
if [ $E -eq 25 -a $G -gt 11 -o $E -eq 24 ] ; then
   E=$(($E + 1))
fi
TAG=$((44 - $E))
if [ $TAG -lt 21 ] ; then
   TAG=$(($TAG + 30))
fi
TAG=$(($TAG + 7 - \( $D + $TAG \) \% 7))
if [ $TAG -gt 31 ] ; then
   TAG=$(($TAG - 31))
```

```
   MON=4
else
   MON=3
fi
echo östern $JAHR ist am ${TAG}.${MON}.\n"
```

4.9.6 Wem die Stunde schlägt...

Wer im Besitz einer Soundkarte ist, kann sich eine schöne Turmuhr, einen Regulator oder Big Ben basteln. Die folgende „Uhr“ hat Stunden- und Viertelstundenschlag. Zuvor ist jedoch ein Eintrag in der `crontab` notwendig:

```
0,15,30,45 * * * * /home/sbin/turmuhr
```

So wird das Skript `turmuhr` jede Viertelstunde aufgerufen. Es werden zwei Sounddateien verwendet, `hour.au` für den Stundenschlag und `quarter.au` für den Viertelstundenschlag. Statt des Eigenbau-Programms *audioplay* kann auch der *sox* verwendet werden oder man kopiert die Dateien einfach nach `/dev/audio`. Die Variable `VOL` steuert die Lautstärke.

```
#!/bin/sh
BELL=/home/local/sounds/hour.au
BELL1=/home/local/sounds/quater.au
PLAY=/usr/bin/audioplay
VOL=60
DATE=‘date +\%H:\%M‘
MINUTE=‘echo $DATE | sed -e ’s/.*://’‘
HOUR=‘echo $DATE | sed -e ’s/:.*//’‘

if [ $MINUTE = 00 ]
then
        COUNT=‘expr \( $HOUR \% 12 + 11 \) \% 12‘
        BELLS=$BELL
        while [ $COUNT != 0 ];
        do
                BELLS="$BELLS $BELL"
                COUNT=‘expr $COUNT - 1‘
        done
        $PLAY -v $VOL -i $BELLS
elif [ $MINUTE = 15 ]
then    $PLAY -v $VOL -i $BELL1
elif [ $MINUTE = 30 ]
then    $PLAY -v $VOL -i $BELL1 $BELL1
elif [ $MINUTE = 45 ]
then    $PLAY -v $VOL -i $BELL1 $BELL1 $BELL1
else
        $PLAY -v $VOL -i $BELL1
fi
```

4.9.7 Eingabe ohne Enter-Taste

Soll nur eine Taste zur Bestätigung gedrückt werden, z. B. `j` oder `n`, lässt sich das mit dem `read`-Kommando nicht realisieren, da die Eingabe immer mit der ⏎-Taste abgeschlossen werden muss. Um eine Eingabe ohne ⏎ zu realisieren, sind zwei Dinge notwendig:

- Schalten des Terminals auf ungepufferte Eingabe
- Einlesen der gewünschten Anzahl von Zeichen

Das Umschalten des Terminals wird mit `stty raw` erreicht. Für die Eingabe wird das `dd`-Kommando (*Disk Dump*) zweckentfremdet. `dd` liest von der Standardeingabe und schreibt auf die Standardausgabe. Für die geplante Aktion werden die Parameter `count` (Anzahl zu lesender Blöcke) und `bs` (*blocksize*) verwendet. `count` enthält die Anzahl der einzulesenden Zeichen, `bs` wird auf `1` gesetzt. Der entstehende Programmteil sieht dann so aus:

```
echo "Alles Loeschen (j/n)\c"
stty raw -echo
INPUT=`dd count=1 bs=1 2> /dev/null`
stty -raw echo
echo $INPUT
case $INPUT in
  j|J) echo "Jawoll" ;;
  n|N) echo "Doch nicht" ;;
  *)   echo "Wat nu?" ;;
esac
```

4.9.8 Ständig kontrollieren, wer sich ein- und ausloggt

```
#!/bin/sh
# PATH=/bin:/usr/bin
NEW=/tmp/WW1.WHO
OLD=/tmp/WW2.WHO
>$OLD                              # OLD neu anlegen
while :                            # Endlosschleife
do
   who >$NEW
   diff $OLD $NEW
   mv $NEW $OLD
    sleep 60
done
```

4.9.9 Optionen ermitteln

Oft ist es wünschenswert, bei Shell-Skripten – auf die gleiche Weise wie bei Programmen – auch Optionen angeben zu können. Die Optionen bestehen aus einem Buchstaben mit dem `-` davor. Bei manchen Optionen folgt auch eine durch die Option spezifizierte Angabe (z. B. beim `pr`-Kommando der Header). Das folgende Fragment zeigt, wie sich solche Optionen behandeln lassen. Die einzige Einschränkung besteht darin, dass sich mehrere Optionen nicht zusammenziehen lassen (`-abc` statt `-a -b -c` geht also nicht). Damit alle Optionen über eine Schleife abgehandelt werden können, wird mit `shift` gearbeitet. Wie üblich, können nach den Optionen noch Dateinamen folgen. Ein Testaufruf könnte lauten:

`otest -a -p` *Parameter* `-c *.txt`

```
#!/bin/sh
# Bearbeiten von Optionen in Shellskripts
# Beispiel: -a -b -c  als einfache Optionen
#           -p <irgend ein Parameter> als "Spezial-Option"
READOPT=0
while [ $READOPT -eq 0 ] ; do     # solange Optionen vorhanden
  case $1 in
      -a) echo öption a"
          shift ;;
      -b) echo öption b"
          shift ;;
      -c) echo öption c"
          shift ;;
      -p) PARAM=$2 ; shift  # Parameter lesen
          echo öption p: $PARAM"
          shift ;;
       *) if ‘echo $1 | grep -s ’^-’‘ ; then # Parm. beginnt mit ’-’
            echo ünknown option $1"
            shift
          else
            READOPT=1  # Ende Optionen, kein shift!
  esac
done
echo "Restliche Parameter : $*"
```

4.9.10 `rename`-Kommando

So mächtig das `mv`-Kommando auch ist, es bietet keine Möglichkeit, eine ganze Reihe von Dateien nach gleichem Schema umzubenennen (z. B. `kap??.txt` zu `kapitel??.txt`). Das folgende Skript leistet dies. Die ersten beiden Parameter enthalten den ursprünglichen Namensteil (z. B. `kap`) und den neuen Namensteil (z. B. `kapitel`). Danach folgt die Angabe der zu bearbeitenden Dateien. Wenn

die Zieldatei schon existiert, wird nicht umbenannt, sondern eine Fehlermeldung ausgegeben.

```
#!/bin/sh
# Alle Dateien umbennen, die durch $3 - $n spezifiziert werden
# dabei wird der String $1 im Dateinamen durch $2 ersetzt,
# wobei auch regulaere Ausdruecke erlaubt sind
if [ $# -lt 3 ] ; then
   echo 'Usage: ren <old string> <new string> files'
   echo 'Example:  ren foo bar *.foo  renames all files'
   echo '             *.foo ---> *.bar'
   exit 1
fi
S1=$1 ; shift
S2=$1 ; shift
while [ $# -gt 0 ]; do
   for OLDF in $1 ; do
      NEWF=`echo $OLDF | sed -e "s/${S1}/${S2}/\qq{
      if [ -f $NEWF ] ; then
           echo "$NEWF exists, $OLDF not renamed"
       else
        echo "renaming $OLDF to $NEWF"
        mv $OLDF $NEWF
      fi
   done
   shift
done
```

4.9.11 Rekursives Suchen in Dateien

Manchmal muss man auf eine ganze Gruppe von Dateien die gleiche Textmanipulation (Suchen und Ersetzen) anwenden, etwa wenn man sich entschlossen hat alle Dateien der eigenen Web-Präsenz endlich mal mit CSS-Styles zu versehen. Das folgende Skript rattert durch einen Dateibaum und führt die Ersetzung bei jeder Datei durch. Der Aufruf von `rsar` (recursive search and replace) benötigt vier Parameter:

```
rsar <start-dir> <file-expression> <search-pattern> <replace-pattern>
```

Das `start-dir` legt fest, welches Verzeichnis rekursiv durchsucht wird, `file-expression` erlaubt die Auswahl bestimmter Dateien (z.B. *.txt) und das Such- und Ersetzungsmuster entspricht dem, was man auch beim `sed` verwenden würde. Hier wird jedoch nicht der `sed`, sondern der `vi` im Kommandozeilenmodus (`ex`) verwendet.

```
#!/bin/sh
PROGNAME=`basename $0`
TEMPDAT=/tmp/`basename $0.$$`
```

```
if test $# -lt 4; then
   echo "$PROGNAME : Recursive search-and-replace-skript."
   echo "usage : $PROGNAME <dir> <file-expr> <search> <replace>"
   echo "Both patterns use ex/vi-syntax !"
else
   find $1 -type f -name "$2" -print > $TEMPDAT
   for NAME in `cat $TEMPDAT`
     do
     echo -n "Processing $NAME.."
     ex $NAME << EOT > /dev/null
1,\$ s/$3/$4/g
wq
EOT
     echo "done."
   done
   rm $TEMPDAT
fi
```

4.9.12 Das Letzte

Das letzte Shell-Programm in diesem Kapitel heißt „Unix-Camping“ und ist nicht ganz ernst zu nehmen:

```
( unzip; strip; touch; finger; mount; gasp; yes; umount; sleep )
```

4.10 Referenz Shell-Programmierung

Die nachfolgende Referenz liefert in alphabetischer Reihenfolge eine kurze Beschreibung der wichtigsten Kommandos zur Shell-Programmierung (ohne jeden Anspruch auf Vollständigkeit!). Bei vielen der hier besprochenen Kommandos handelt es sich um eingebaute `bash`-Kommandos, zum Teil aber auch um reguläre Linux-Kommandos, die sich besonders zur Anwendung bei der Shell-Programmierung eignen.

Variablenverwaltung und Parameterauswertung

`alias`	definiert eine Abkürzung
`declare`	definiert eine (Umgebungs-)Variable
`export`	definiert eine Umgebungsvariable
`local`	definiert lokale Variablen in einer Funktion
`read`	liest eine Variable ein
`readonly`	zeigt alle schreibgeschützten Variablen an
`shift`	verschiebt die Parameterliste
`unalias`	löscht eine Abkürzung

`unset`	löscht eine Variable

Umgang mit Zeichenketten

`basename`	ermittelt den Dateinamen eines Pfads
`dirname`	ermittelt das Verzeichnis eines Pfads
`expr`	führt einen Mustervergleich durch

Verzweigungen, Schleifen

`break`	beendet eine Schleife vorzeitig
`case`	leitet eine Fallunterscheidung ein
`continue`	überspringt den Schleifenkörper
`exit`	beendet das Shell-Programm
`for`	leitet eine Schleife ein
`function`	definiert eine neue Funktion
`if`	leitet eine Verzweigung ein
`local`	definiert lokale Variablen in einer Funktion
`return`	beendet eine Funktion
`source`	führt die angegebene Shell-Datei aus
`test`	wertet eine Bedingung aus
`until`	leitet eine Schleife ein
`while`	leitet eine Schleife ein

Ausgabe

`cat`	gibt eine Datei auf dem Bildschirm aus
`echo`	gibt eine Zeile Text aus
`printf`	ermöglicht eine formatierte Ausgabe wie unter C
`setterm`	verändert die Schriftart, löscht den Bildschirm

Sonstiges

`dialog`	zeigt eine Dialogbox an
`dirs`	zeigt die Liste der gespeicherten Verzeichnisse an
`eval`	wertet das angegebene Kommando aus
`file`	versucht, den Typ einer Datei festzustellen
`popd`	wechselt in das letzte gespeicherte Verzeichnis
`pushd`	speichert das aktuelle und wechselt in ein neues Verzeichnis
`sleep`	wartet eine vorgegebene Zeit
`trap`	führt beim Eintreten eines Signals ein Kommando aus
`ulimit`	kontrolliert die von der Shell beanspruchten Ressourcen
`wait`	wartet auf das Ende eines Hintergrundprozesses

alias `abkürzung='kommando'`

`alias` definiert eine Abkürzung.

basename `zeichenkette [endung]`

`basename` liefert den Dateinamen des übergebenen Pfads. `basename /usr/man /man1/gnroff.1` führt also zum Ergebnis `gnroff.1`. Wenn als zusätzlicher Parameter eine Dateiendung angegeben wird, so wird diese (falls vorhanden) aus dem Dateinamen entfernt.

break `[` *n*`]`

`break` bricht eine `for`-, `while`- oder `until`-Schleife vorzeitig ab. Das Shell-Programm wird beim nächsten Kommando nach dem Schleifenende fortgesetzt. Durch die Angabe eines optionalen Zahlenwerts können *n* Schleifenebenen abgebrochen werden.

case *ausdruck* `in`
    muster1 `)` *kommandos*`;;`
    muster2 `)` *kommandos*`;;`
    ...
esac

`case` wird zur Bildung von Mehrfachverzweigungen verwendet, wobei als Kriterium für die Verzweigung eine Zeichenkette angegeben wird (zumeist eine Variable oder ein Parameter, der dem Shell-Programm übergeben wird). Diese Zeichenkette wird der Reihe nach mit den Mustern verglichen, wobei in diesen Mustern die Jokerzeichen für Dateinamen (`*?[]`) verwendet werden können. In einem `case`-Zweig können auch mehrere durch `|` getrennte Muster angegeben werden.

Sobald ein Muster zutrifft, werden die Kommandos ausgeführt, die zwischen der runden Klammer `)` und den beiden Semikola folgen. Anschließend wird das Programm nach `esac` fortgesetzt.

cat `<< ende`

`cat` liest in dieser Syntaxvariante so lange Text der aktuellen Shell-Datei und zeigt diesen auf dem Bildschirm an, bis die Zeichenkette `ende` auftritt. `cat` kann damit zur bequemen Ausgabe größerer Textmengen verwendet werden, ohne für jede Zeile ein `echo`-Kommando ausführen zu müssen. Die Syntax `<< ende` kann aber natürlich auch für alle anderen Kommandos verwendet werden, um Text bis zur Zeichenkette `ende` aus der aktuellen Datei zu lesen.

continue `[n]`

`continue` überspringt den Körper einer `for`-, `while`- oder `until`-Schleife und setzt die Schleife mit dem nächsten Durchlauf fort. Durch den optionalen Zahlenwert kann dieser Vorgang auch für äußere Schleifenebenen durchgeführt werden.

dialog `[--clear] [--title text] [dialogoptionen]`

`dialog` zeigt einen Dialog an. Das Kommando hat genau genommen nichts mit der `bash` zu tun, sondern ist ein eigenständiges Programm zur textorientierten Ein- und Ausgabe von Daten. Es kann dazu verwendet werden, um `bash`-Scripts trotz der Einschränkungen durch den Textmodus ein etwas moderneres Aussehen zu geben.

Die Optionen `--clear` und `--title` können mit jeder der angegebenen Dialogoptionen kombiniert werden. `--clear` bewirkt, dass der Dialog nach seinem Ende von der Konsole entfernt wird. (Die gesamte Konsole ist anschließend blau! Führen Sie eventuell anschließend `setterm -clear` aus, um die Default-Farben wiederherzustellen.) `--title` ermöglicht die Einstellung des Dialogtitels. `dialog` kennt die folgenden Dialogformen:

- **--msgbox text höhe breite** zeigt einen Meldungstext an, der mit (↵) bestätigt werden muss.
- **--infobox text höhe breite** wie oben, aber ohne Bestätigung. Das Programm wird sofort fortgesetzt. Die Dialogbox bleibt so lange auf dem Bildschirm stehen, bis dieser gelöscht wird.
- **--yesno text höhe breite** zeigt eine Dialogbox für eine Ja/Nein-Entscheidung an.
- **--inputbox text höhe breite** ermöglicht die Eingabe einer Textzeile.
- **--textbox datei höhe breite** zeigt die Textdatei an (ohne Editiermöglichkeit, aber mit Scrolling).
- **--menu text h b menühöhe menüpkt1 menütext1 mp2 mt2 ...** ermöglicht die Auswahl einer Option (eines Menüpunkts).
- **--checklist text höhe breite listenhöhe**
- **option1 text1 status1 option2 text2 status2 ...** a ermöglicht die gleichzeitige Auswahl mehrerer Optionen.

`dialog` liefert als Rückgabewert 0, wenn der Dialog mit OK oder YES beendet wurde, 1, wenn der Dialog mit CANCEL oder NO beendet wurde, oder 255, wenn der Dialog mit (Esc) beendet wurde. Bei den Dialogtypen, die einen Text, Menütitel oder eine Liste von Optionen als Ergebnis liefern, werden die Resultate in die Standardfehlerdatei geschrieben. Im Normalfall ist daher eine Umleitung in eine temporäre Datei mit `2> tmp` erforderlich. Nach dem Ende der Eingabe kann diese Datei ausgewertet werden.

dirname `zeichenkette`

`dirname` liefert den Pfad eines vollständigen Dateinamens. `dirname /usr/bin/groff` liefert also `/usr/bin`.

dirs

`dirs` zeigt die Liste der durch `pushd` gespeicherten Verzeichnisse an.

echo `[optionen] zeichenkette`

`echo` gibt die angegebene Zeichenkette auf dem Bildschirm aus. Die Zeichenkette sollte in doppelte oder einfache Hochkommata gestellt werden.

-e beachtet unter anderem die Sonderzeichen `\a` (Beep), `\n` (Zeilenende) und `\t` (Tabulator) in der Zeichenkette. Durch `echo -e "\a"` kann also ein Warnton ausgegeben werden.

-n wechselt beim Ende der Ausgabe nicht in eine neue Zeile. Die Ausgabe kann durch eine weitere `echo`-Anweisung fortgesetzt werden.

eval `$var`

`eval` interpretiert den Inhalt der Variablen als Kommandozeile, wertet diese Zeile aus (mit allen bekannten Substitutionsmechanismen) und führt das Kommando schließlich aus. `eval` ist immer dann erforderlich, wenn ein in einer Variablen gespeichertes Kommando ausgeführt werden soll und dieses Kommando diverse Sonderzeichen der Shell enthält.

Erst mit der Verwendung von `eval` kann das in der Variablen `kom` gespeicherte Kommando ausgeführt werden. Der erste Versuch, das Kommando auszuführen, scheitert, weil die `bash` das Pipe-Zeichen `|` nicht mehr auswertet, nachdem sie `$kom` durch seinen Inhalt ersetzt hat.

exec `kommando`

`exec` startet das angegebene Kommando als Ersatz zur laufenden `bash`. Das Kommando kann beispielsweise dazu verwendet werden, eine andere Shell zu starten. Die laufende Shell wird dadurch auf jeden Fall beendet. (Bei einem normalen Kommandostart bleibt die `bash` im Hintergrund aktiv, bis das Kommando beendet ist.)

exit `[rückgabewert]`

`exit` beendet ein Shell-Programm. Wenn kein Rückgabewert angegeben wird, gibt das Programm 0 (ok) zurück.

export `[optionen] variable [=wert]`

`export` deklariert die angegebene Shell-Variable als Umgebungs-Variable. Damit ist die Variable auch in allen aufgerufenen Kommandos und Subshells verfügbar. Optional kann dabei auch eine Variablenzuweisung erfolgen. Wenn das Kommando ohne Parameter aufgerufen wird, werden alle Umgebungs-Variablen angezeigt.

-n macht eine Umgebungs-Variable wieder zu einer normalen Shell-Variablen. Das Kommando hat damit genau die umgekehrte Wirkung wie bei der Verwendung ohne Optionen.

expr `zeichenkette : muster`

`expr` kann zur Auswertung arithmetischer Ausdrücke, zum Vergleich zweier Zeichenketten etc. eingesetzt werden. Von `test` und dem Substitutionsmechanismus `$[...]` unterscheidet sich `expr` insofern, als es in der oben angeführten Syntaxvariante Mustervergleiche für Zeichenketten durchführen kann (reguläre Ausdrücke).

file `[optionen] datei`

`file` versucht festzustellen, welchen Datentyp die als Parameter angegebene Datei hat. Als Ergebnis liefert `file` eine Zeichenkette mit dem Dateinamen und dem Typ der Datei. *Achtung*: Textdateien mit deutschen Sonderzeichen werden nicht als Textdateien klassifiziert, sondern als `data`. Die Klassifizierung basiert auf der Datei `/etc/magic`. `file -z datei` versucht den Datentyp einer komprimierten Datei zu erkennen.

for *var* [**in** *liste*;] **do**
  kommandos
done

`for` bildet eine Schleife. In die angegebene Variable werden der Reihe nach alle Listenelemente eingesetzt. Die Liste kann auch mit Jokerzeichen für Dateinamen oder mit `{...}`-Elementen zur Zusammensetzung von Dateinamen gebildet werden. Wenn auf die Angabe der Liste verzichtet wird, durchläuft die Variable alle der Shell-Datei übergebenen Parameter (also `in $*`).

[**function**] *name*()
  { *kommandos* }

`function` definiert eine Subfunktion, die innerhalb der Shell-Datei wie ein neues Kommando aufgerufen werden kann. Innerhalb der Funktion können mit `local` lokale Variablen definiert werden. Funktionen können rekursiv aufgerufen werden. Den einzelnen Funktionen können Parameter wie Kommandos übergeben werden. Innerhalb der Funktion können diese Parameter den Variablen `$1`, `$2` etc. entnommen werden.

if *bedingung*; **then**
  kommandos
[**elif** *bedingung*; **then**
  kommandos]
[**else**

  kommandos]

fi

Das `if`-Kommando leitet eine Verzweigung ein. Der Block nach `then` wird nur ausgeführt, wenn die Bedingung erfüllt ist. Andernfalls werden (beliebig viele, optionale) `elif`-Bedingungen ausgewertet. Gegebenenfalls wird der ebenfalls optionale `else`-Block ausgeführt.

Als Bedingung können mehrere Kommandos angegeben werden. Nach dem letzten Kommando muss ein Semikolon folgen. Als Kriterium gilt der Rückgabewert des letzten Kommandos. Vergleiche und andere Tests können mit dem Kommando `test` durchgeführt werden. Statt `test` ist auch eine Kurzschreibweise in eckigen Klammern zulässig, dabei muss aber nach `[` und vor `]` jeweils ein Leerzeichen stehen.

local `var[=wert]`

`local` definiert eine lokale Variable. Das Kommando kann nur innerhalb einer selbst definierten Funktion verwendet werden (siehe `function`). Vor und nach dem Gleichheitszeichen dürfen keine Leerzeichen angegeben werden.

popd

`popd` wechselt in ein zuvor mit `pushd` gespeichertes Verzeichnis zurück. Das Verzeichnis wird aus der Verzeichnisliste entfernt.

printf `format para1 para2 para3 ...`

`printf` erlaubt es, Ausgaben in der Syntax des C-Kommandos *printf* zu formatieren. Detaillierte Informationen zu den Formatierungsmöglichkeiten erhalten Sie mit `man 3 printf`.

pushd `verzeichnis`

`pushd` speichert das aktuelle Verzeichnis und wechselt anschließend in das angegebene Verzeichnis. Mit `popd` kann in das ursprüngliche Verzeichnis zurückgewechselt werden. `dirs` zeigt die Liste der gespeicherten Verzeichnisse an.

read `[var1 var2 var3 ...]`

`read` liest eine Zeile Text in die angegebenen Variablen. `read` erwartet die Daten aus der Standardeingabe. Wenn keine Variable angegeben wird, schreibt `read` die Eingabe in die Variable `REPLY`. Wenn genau eine Variable angegeben wird, schreibt `read` die gesamte Eingabe in diese eine Variable. Wenn mehrere Variablen angegeben werden, schreibt `read` das erste Wort in die erste Variable, das zweite Wort in die zweite Variable ... und den verbleibenden Rest der Eingabe in die letzte Variable. Wörter werden dabei durch Leer- oder Tabulatorzeichen getrennt.

Das `read`-Kommando sieht keine Möglichkeit vor, einen Infotext als Eingabeaufforderung auszugeben. Deswegen ist es zweckmäßig, den Anwender vor der Ausführung von `read`-Kommandos mit `echo -n` über den Zweck der Eingabe zu informieren.

setterm `[option]`

`setterm` verändert diverse Einstellungen des Terminals. Wenn das Kommando ohne die Angabe einer Option ausgeführt wird, zeigt es eine Liste aller möglichen Optionen an. Nützliche Optionen zur Shell-Programmierung sind:

- **-bold on** | **off** aktiviert bzw. deaktiviert die fette Schrift. In Textkonsolen erscheint der Text zwar nicht fett, aber immerhin in einer anderen Farbe als der sonstige Text.
- **-clear** löscht den Inhalt des Terminals.
- **-default** stellt Farben und Textattribute auf die Default-Einstellung zurück.
- **-half-bright on** | **off** stellt hervorgehobene Schrift ein/aus.
- **-reverse on** | **off** stellt inverse Schrift ein/aus.
- **-underline on** | **off** stellt unterstrichene Schrift ein/aus.

shift `[n]`

`shift` schiebt die dem Shell-Programm übergebene Parameterliste durch die vordefinierten Variablen `$1` bis `$9`. Wenn `shift` ohne Parameter verwendet wird, werden die Parameter um eine Position verschoben, andernfalls um *n* Positionen. `shift` ist besonders dann eine wertvolle Hilfe, wenn mehr als neun Parameter angesprochen werden sollen. *Achtung*: Einmal mit `shift` aus den Variablen geschobene Parameter können nicht mehr angesprochen werden. Sie werden auch aus der Variablen `$*` entfernt.

sleep `zeit`

`sleep` versetzt das laufende Programm für die angegebene Zeit in den Ruhezustand. Das Programm konsumiert in dieser Zeit praktisch keine Rechenzeit. Die Zeitangabe erfolgt normalerweise in Sekunden. Optional können die Buchstaben `m`, `h` oder `d` angehängt werden, um die Zeit in Minuten, Stunden oder Tagen anzugeben.

source `datei`

`source` führt die angegebene Shell-Datei so aus, als befänden sich die darin enthaltenen Kommandos an der Stelle des `source`-Kommandos. Nach der Ausführung der Datei wird das laufende Shell-Programm in der nächsten Zeile fortgesetzt. Zur Ausführung der angegebenen Datei wird keine neue Shell gestartet. Alle Variablen (inklusive der Parameterliste) gelten daher auch für die angegebene Datei. Wenn in dieser Datei `exit` ausgeführt wird, kommt es *nicht* zu einem Rücksprung in das Programm mit dem `source`-Kommando, sondern zu einem sofortigen Ende der Programmausführung. Zu `source` existiert die Kurzform `.␣datei`.

`test ausdruck`

`test` wird zur Formulierung von Bedingungen verwendet und zumeist in `if`-Abfragen und Schleifen eingesetzt. Je nachdem, ob die Bedingung erfüllt ist, liefert es den Wahrheitswert 0 (wahr) oder 1 (falsch). Statt `test` kann auch die Kurzschreibweise `[␣ausdruck␣]` verwendet werden, wobei Leerzeichen vor und nach dem Ausdruck angegeben werden müssen.

Wenn *test* oder die Kurzschreibweise *[ausdruck]* als Bedingung in einer Verzweigung oder Schleife verwendet wird, muss die Bedingung mit einem Semikolon abgeschlossen werden, also z. B. `if [ "$1" = "abc" ]; then ...`

if-Abfragen können manchmal durch die Formulierung `test "$1" = "abc" && kommando` ersetzt werden. In diesem Fall ist kein Semikolon erforderlich. Das Kommando wird nur ausgeführt, wenn die vorherige Bedingung erfüllt war.

Zeichenketten

`[` *zk* `]`	wahr, wenn die Zeichenkette nicht leer ist
`[ -z` *zk* `]`	wahr, wenn die Zeichenkette leer ist (0 Zeichen)
`[` *zk1* `=` *zk2* `]`	wahr, wenn die Zeichenketten übereinstimmen
`[` *zk1* `!=` *zk2* `]`	wahr, wenn die Zeichenketten voneinander abweichen

Die Zeichenketten bzw. Variablen sollten in Hochkommata gestellt werden (z. B. `[ "$1" = "abc" ]` oder `[ "$a" = "$b" ]`). Andernfalls kann es bei Zeichenketten mit mehreren Wörtern zu Fehlern kommen.

Zahlen

`[` *z1* `-eq` *z2* `]`	wahr, wenn die Zahlen gleich sind (equal)
`[` *z1* `-ne` *z2* `]`	wahr, wenn die Zahlen ungleich sind (not equal)
`[` *z1* `-gt` *z2* `]`	wahr, wenn *z1* größer *z2* ist (greater than)
`[` *z1* `-ge` *z2* `]`	wahr, wenn *z1* größer gleich *z2* ist (greater equal)
`[` *z1* `-lt` *z2* `]`	wahr, wenn *z1* kleiner *z2* ist (less than)
`[` *z1* `-le` *z2* `]`	wahr, wenn *z1* kleiner gleich *z2* ist (less equal)

Dateien (auszugsweise)

`[ -d` *dat* `]`	wahr, wenn es sich um ein Verzeichnis handelt (directory)
`[ -e` *dat* `]`	wahr, wenn die Datei existiert (exist)
`[ -f` *dat* `]`	wahr, wenn es sich um eine einfache Datei (und nicht um ein Device, ein Verzeichnis ...) handelt (file)
`[ -L` *dat* `]`	wahr, wenn es sich um einen symbolischen Link handelt
`[ -r` *dat* `]`	wahr, wenn die Datei gelesen werden darf (read)
`[ -s` *dat* `]`	wahr, wenn die Datei mindestens 1 Byte lang ist (size)
`[ -w` *dat* `]`	wahr, wenn die Datei verändert werden darf (write)
`[ -x` *dat* `]`	wahr, wenn die Datei ausgeführt werden darf (execute)
`[` *dat1* `-ef` *dat2* `]`	wahr, wenn beide Dateien denselben I-Node haben (equal file)
`[` *dat1* `-nt` *dat2* `]`	wahr, wenn Datei 1 neuer als Datei 2 ist (newer than)

Verknüpfte Bedingungen

`[ !` *bed* `]`	wahr, wenn die Bedingung nicht erfüllt ist
`[` *bed1* `-a` *bed2* `]`	wahr, wenn beide Bedingungen erfüllt sind (and)
`[` *bed1* `-o` *bed2* `]`	wahr, wenn mindestens eine der Bedingungen erfüllt ist (or)

trap `[kommando] n`

`trap` führt das angegebene Kommando aus, wenn in der `bash` das angegebene Signal auftritt. Wenn kein Kommando angegeben wird, ignoriert das Programm bzw. die `bash` das betreffende Signal. `trap -l` liefert eine Liste aller möglichen Signale und der ihnen zugeordneten Kenn-Nummern.

unalias `abkürzung`

`unalias` löscht eine vorhandene Abkürzung. Wenn das Kommando mit der Option `-a` aufgerufen wird, löscht es alle bekannten Abkürzungen.

unset `variable`

`unset` löscht die angegebene Variable.

until *bedingung*; **do**
    kommandos
done

`until` dient zur Bildung von Schleifen. Die Schleife wird so lange ausgeführt, wie die angegebene Bedingung erfüllt ist. Das Schleifenkriterium ist der Rückgabewert des Kommandos, das als Bedingung angegeben wird. Vergleiche und Tests werden mit dem Kommando `test` oder dessen Kurzform in eckigen Klammern durchgeführt.

wait `[prozessnummer]`

`wait` wartet auf das Ende des angegebenen Hintergrundprozesses. Wenn keine Prozessnummer angegeben wird, wartet das Kommando auf das Ende aller laufenden, von der Shell gestarteten Hintergrundprozesse.

while *bedingung*; **do**
    kommandos
done

`while` dient zur Bildung von Schleifen. Die Schleife wird so lange ausgeführt, bis die angegebene Bedingung zum ersten Mal nicht mehr erfüllt ist. Das Schleifenkriterium ist der Rückgabewert des Kommandos, das als Bedingung angegeben wird. Vergleiche und Tests werden mit dem Kommando `test` oder dessen Kurzform in eckigen Klammern durchgeführt.

4.11 Referenz aller Sonderzeichen

- `;` trennt mehrere Kommandos
- `:` Shell-Kommando, das nichts tut
- `.` Shell-Programm ohne eigene Subshell starten (`. datei`) (entspricht `source datei`)
- `#` leitet einen Kommentar ein
- `#!/bin/sh` identifiziert die gewünschte Shell für das Shell-Programm
- `&` führt das Kommando im Hintergrund aus (`kom &`)
- `&&` bedingte Kommando-Ausführung (`kom1 && kom2`)
- `&>` Umleitung von Standardausgabe und -fehler (entspricht `>&`)
- `|` bildet Pipes (`kom1 | kom2`)
- `||` bedingte Kommando-Ausführung (`kom1 || kom2`)
- `*` Jokerzeichen für Dateinamen (beliebig viele Zeichen)
- `?` Jokerzeichen für Dateinamen (ein beliebiges Zeichen)
- `[abc]` Jokerzeichen für Dateinamen (ein Zeichen aus `abc`)
- `[ ausdruck ]` Kurzschreibweise für `test ausdruck`
- `(...)` Kommandos in derselben Shell ausführen (`(kom1; kom2)`)
- `{...}` Kommandos gruppieren
- `~` Abkürzung für das Heimatverzeichnis
- `>` Ausgabeumleitung in eine Datei (`kom > dat`)
- `>>` Ausgabeumleitung; an vorhandene Datei anhängen
- `>&` Umleitung von Standardausgabe und -fehler (entspricht `&>`)
- `2>` Umleitung der Standardfehlerausgabe
- `<` Eingabeumleitung aus einer Datei (`kom < dat`)
- `<< ende` Eingabeumleitung aus der aktiven Datei bis zu `ende`
- `$` Kennzeichnung von Variablen (`echo $var`)
- `$!` PID des zuletzt gestarteten Hintergrundprozesses
- `$$` PID der aktuellen Shell

`$0`	Dateiname des gerade ausgeführten Shell-Scripts
`$1` bis `$9`	die ersten neun dem Kommando übergebenen Parameter
`$#`	Anzahl der dem Shell-Programm übergebenen Parameter
`$*` oder `$@`	Gesamtheit aller übergebenen Parameter
`$?`	Rückgabewert des letzten Kommandos (0 = OK oder Fehlernummer)
`$(...)`	Kommandosubstitution (`echo $(ls)`)
`${...}`	diverse Spezialfunktionen zur Bearbeitung von Zeichenketten
`$[...]`	arithmetische Auswertung (`echo $[2+3]`)
`"..."`	Auswertung der meisten Sonderzeichen verhindern
`'...'`	Auswertung aller Sonderzeichen verhindern
blabla	nur als Trenner
`` `...` ``	Kommandosubstitution (`` echo `ls` ``)
`\`*zeichen*	hebt die Wirkung des Sonderzeichens auf

4.12 Aufgaben

1. Es geht um Kommandosubstitution. Was beinhaltet nach Abarbeitung der Kommandofolge die Datei `datei3`?

```
echo "blau" > kunde
echo "gruen" > kun.de
echo "gelb" >> kunde
echo "rot" > kunde.r
echo kund* >datei2
cat `cat datei2` > datei3
```

2. Mit Pipes kann man recht komfortable Kommandos bilden. Verwenden Sie die Hintereinanderschaltung von `ps -ef` und `grep`, um die Prozesse eine bestimmten Benutzers (z. B. root) oder eines bestimmten Terminals anzuzeigen.

3. Geben Sie einen Aufruf von `find` an, der im Verzeichnis `/home` alle Dateien sucht, auf die innerhalb der letzten 10 Tage zugegriffen wurde.

4. Welches `find`-Kommando müsste man verwenden, um in allen Dateien auf der Platte mit der Endung „.txt“ nach der Zeichenkette „UNIX“ zu suchen.

5. Schreiben Sie ein Shell-Skript, das in einer Endlosschleife alle 10 Sekunden die Uhrzeit (Stunde und Minute) per `banner`-Kommando auf den Bildschirm ausgibt und vorher den Bildschirm löscht.

6. Schreiben Sie ein Shell-Skript „ggt“, daß den größten gemeinsamen Teiler der als Parameter übergebenen beiden Zahlen berechnet. Formulieren Sie die Berechnung des GGT als Shell-Funktion. Der Algorithmus lautet folgendermaßen:

```
ggt(x,y):
solange x ungleich y ist, wiederhole
  falls x > y  dann  x = x - y
  sonst              y = y - x;
```

Denken Sie daran, daß man zum Rechnen das Kommando `expr` braucht.

7. Eine Datei namens `personal` habe folgendes Aussehen:

```
[Name]        [Vorname]     [Wohnort]     [Geb.-Datum]
=======================================================
Meyer         Peter         Berlin         10.10.1970
Schulze       Axel          Hamburg        12.12.1980
Lehmann       Rita          München        17.04.1971
```

   Sortieren Sie die Daten der Datei (ohne Zeile 1 und 2) nach dem Namen und geben Sie das Ergebnis in eine Datei `personal.sort` aus. Die beiden Überschriftszeilen sollen natürlich wieder am Dateianfang stehen.

8. Schreiben Sie ein Shellscript, das an alle Benutzer mit der Gruppennummer 100 eine E-Mail verschickt. Betreff und die Datei, welche den E-Mailtext enthält, werden als Parameter an das Script übergeben.

Kapitel 5

WWW, E-Mail, NFS, SSH und SCP

Einführung

Dieses Kapitel stellt die wichtigsten Programme zum Surfen im Web, zum Lesen und Verfassen von E-Mails sowie zum Dateizugriff und -transfer vor. Unter Linux steht eine breite Palette derartiger Programme zur Verfügung. Die folgende Liste gibt Ihnen einen ersten Überblick über die wichtigsten Programme:
Mozilla: Universalprogramm für Web, E-Mail und News
Firefox: Webbrowser (hervorgegangen aus Mozilla)
Thunderbird: Mail-Client (hervorgegangen aus Mozilla)
Konqueror: KDE-Standardbrowser
Epiphany: Gnome-Standardbrowser
KMail: KDE-Mail-Client
Lynx: textbasierter Browser
Mutt: textbasierter Mail-Client

5.1 Webbrowser

Dieser Abschnitt stellt die wichtigsten unter Linux verfügbaren Webbrowser vor. Da wir davon ausgehen, dass Sie die Grundfunktionen dieser Programme auch ohne dieses Buch nutzen können, beschränken wir uns auf die Beschreibung von Besonderheiten bzw. Konfigurationsdetails.

5.1.1 Die Mozilla-Familie

Mozilla hat sich in den vergangenen Jahren zu dem Standardbrowser für Linux entwickelt. Mozilla setzt die verschiedenen Web- und HTML-Standards sehr ge-

nau um und verfügt über einen ausgereiften JavaScript-Interpreter. Die Anzahl der Websites, die mit Mozilla nicht genutzt werden können, ist daher ziemlich gering. Meist handelt es sich dabei um Sites, die nicht standardisierte, Internet-Explorer-spezifische Erweiterungen verwenden oder die Plugins bzw. ActiveX-Komponenten nutzen, die nur unter Microsoft Windows zur Verfügung stehen.

Mozilla ist ein Universalprogramm, das gleichermaßen zum Surfen im Web, zur Verwaltung von E-Mails und zum Lesen von News eingesetzt werden kann. Je nach Installation kann das Programm auch als IRC-Client, als HTML-Editor und zur Terminverwaltung verwendet werden (siehe *Tools*- bzw. *Fenster*-Menü).

Firefox und Thunderbird: Anfang 2003 beschlossen die Mozilla-Entwickler, das immer größer werdende Programm in zwei wesentliche Komponenten zu zerlegen: den neuen Webbrowser „Mozilla Firebird“ und den neuen E-Mail-Client „Mozilla Thunderbird“. Anfang 2004 wurde „Firebird“ in „Firefox“ umbenannt. Die beiden Programme haben zwar noch einige Ecken und Kanten (insbesondere Firefox), sind aber schon gut verwendbar. Es wird aber wohl noch etwas dauern, bis beide Programme wirklich ausgereift sind. Aus diesem Grund wird auch Mozilla selbst weiterentwickelt. In den kommenden Mozilla-Versionen sind allerdings kaum neue Funktionen zu erwarten. Ausführliche Informationen über alle drei Projekte finden Sie ausgehend von der Mozilla-Website *http://www.mozilla.org/*.

Netscape wird auch oft im Zusammenhang mit Mozilla genannt. Tatsächlich sind diese beiden Programme eng verwandt und nutzen dieselbe Code-Grundlage. Die wohl wichtigste gemeinsame Komponente ist die so genannte Render-Engine Gecko. Gecko ist der Programmteil, der für die Darstellung von HTML-Seiten am Bildschirm zuständig ist.

Das Programm Netscape ist wie Mozilla kostenlos erhältlich. Netscape enthält allerdings zusätzliche Funktionen, zu denen der Quelltext nicht verfügbar ist, insbesondere den AOL Instant Messenger. Davon abgesehen unterscheiden sich Mozilla und Netscape weder in ihrem Aussehen noch in ihrer Bedienung nennenswert. Bei den meisten Linux-Distributionen wird nur Mozilla mitgeliefert, weswegen auch dieses Kapitel nur Mozilla behandelt.

Besonderheiten und Konfigurationsmöglichkeiten

Spracheinstellung: Per Default erscheint das Menü von Mozilla oft in Englisch. Wenn Sie deutsche Menü-Einträge vorziehen, können Sie in *Edit* → *Settings* → *Appearance* → *Language* ein anderes Sprachpaket aktivieren. Falls nur das englische Sprachpaket installiert ist, können Sie aus dem Optionendialog andere Sprachpakete herunterladen und installieren.

Schriftgröße: Wenn Sie einen hochauflösenden Monitor benutzen, ist es empfehlenswert, die Schrift zu vergrößern. Im verschachtelten Konfigurationsdialog

Bearbeiten → *Einstellungen* finden Sie Einstellmöglichkeiten für die Schriftdarstellung im Dialogblatt *Erscheinungsbild* → *Schriftarten*. Im Dialog können Sie auch Default-Schriften für verschiedene Schriftarten auswählen. Hier sollten Sie auf TrueType-Schriften zurückgreifen, die auf Ihrem Rechner installiert sind. Sie erreichen damit eine optimale Darstellungsqualität.

URLs per Mausklick einfügen: (ab Mozilla 1.4) Wenn Sie eine Webadresse in einem beliebigen Linux-Programm mit der Maus markiert haben, können Sie den Inhalt der Zwischenablage durch einen Klick mit der mittleren Maustaste in die Adressleiste von Mozilla einfügen. Das wäre noch nicht weiter bemerkenswert. Wenn Sie den Mausklick aber genau über dem Icon innerhalb der Adressleiste durchführen, dann ersetzt der Inhalt der Zwischenablage die aktuelle Adresse; außerdem wird die neue Adresse sofort geladen. Auf diese Weise können Sie also äußerst effizient zu einer anderen Webadresse wechseln.

Link- und Textsuche: Einige Mozilla-Funktionen lassen sich besonders effizient per Tastatur nutzen. So starten Sie mit ', A, B, C die Suche nach einem Link, der mit den Buchstaben *abc* beginnt. Je nach Einstellung in *Erweitert* → *Navigation mit der Tastatur* funktioniert die Linksuche auch ohne die vorherige Eingabe von '.

Analog zur Linksuche können Sie innerhalb der aktuellen Seite mit /, A, B, C nach einem beliebigen Text suchen, der mit *abc* beginnt. Strg+G wiederholt die Suche.

Sidebar: Zur Verwaltung der Bookmarks, aber auch für eine Reihe von Zusatzfunktionen dient eine so genannte Sidebar, die mit F9 rasch ein- und ausgeblendet werden kann. Die Inhalte der Sidebar können Sie mit *Tabs* → *Sidebar anpassen* organisieren.

Internet-Suche: Ausgesprochen praktisch ist auch das Sidebar-Element zur Suche. Per Default wird zur Suche Google verwendet, es sind aber einige weitere Such-Engines vorkonfiguriert. Die Links zu den Suchergebnissen werden innerhalb der Sidebar dargestellt. Wenn Sie im Konfigurationsdialog *Navigator* → *Internet-Suche* den erweiterten Modus aktivieren, können Sie auch selbst eine neue Such-Engine definieren.

Formulardaten und Login-Passwörter: Mozilla kann sich wiederkehrende Formulardaten und Login-Passwörter merken. Die entsprechenden Einstellmöglichkeiten finden Sie im Konfigurationsdialog unter *Datenschutz* → *Formulare* bzw. - → *Passwörter*.

JavaScript und Popup-Fenster: JavaScript ist per Default für den Webbrowser aktiv (nicht aber die Darstellung von E-Mails und News-Beiträgen). Diverse JavaScript-Einstellungen können Sie im Einstellungsdialog *Erweitert* → *Skripte* durchführen, beispielsweise Popup-Fenster verbieten (*Datenschutz* → *Popup-Fenster*).

Lokaler Cache: Mozilla verwaltet einen lokalen Zwischenspeicher, in dem zuletzt besuchte Webseiten, Bilder etc. gespeichert werden. Im Dialog *Erweitert* → *Cache* können Sie die Cache-Größe einstellen bzw. den Cache löschen. Im

Hauptfenster führt die Adresse *about:cache* zu einer Liste aller Dateien, die momentan zwischengespeichert sind.

Proxy: Wenn in Ihrem lokalen Netz der Webzugang nur über einen Proxy-Server möglich ist, müssen Sie die Adresse des Proxy-Servers im Konfigurationsdialog *Erweitert* → *Proxies* angeben.

Drucken: Mozilla bietet per Default keine Möglichkeit zur Druckerauswahl. Die einfachste Abhilfe für diese Einschränkung besteht darin, dass Sie im Druckdialog bei den Druckereigenschaften ein externes Programm angeben, in dem Sie den Drucker auswählen können (z. B. `kprinter` (KDE), `g-print` (Gnome) oder `xpp`). Das führt dazu, dass Mozilla die zu druckenden PostScript-Daten an das betreffende Programm weiterleitet. Dort können Sie den gewünschten Drucker auswählen.

Wenn Sie sich mit den schwachen Druckfunktionen von Mozilla nicht begnügen wollen, können Sie das Zusatzprogramm `xprint` installieren. Hierbei handelt es sich um ein Hintergrundprogramm (Dämon), das im Rahmen des Init-V-Prozesses gestartet werden muss. Mozilla kommuniziert mit diesem Programm, das wiederum die Daten an das Drucksystem weiterleitet. Sobald `xprint` erfolgreich installiert ist, zeigt der Mozilla-Druckdialog alle zur Auswahl stehenden Drucker an. Weitere Informationen zu `xprint` finden Sie unter *http://www.mozilla.org/projects/xprint/*.

MIME (*Multipurpose Internet Mail Extensions*) sagt dem Webbrowser, welches Programm er zur Verarbeitung der Datei `name.xyz` verwenden soll. Die Mozilla-MIME-Einstellungen erfolgen im Konfigurationsdialog *Navigator* → *Hilfsanwendungen*. Wenn Sie für einen neuen Dateityp MIME-Informationen speichern möchten, müssen Sie drei Informationen angeben: den MIME-Typ (z. B. *application/postscript*), die Dateikennung (z. B. `ps` für `*.ps`-Dateien) und den exakten Dateinamen des Programms, mit dem die Datei geöffnet werden soll.

Tipp

Wenn Sie einen Link anklicken, dessen Dateityp nicht in den MIME-Einstellungen enthalten ist, erscheint ein Dialog, in dem Sie die Datei entweder speichern (Download) oder mit einem Programm öffnen können. Zum Öffnen müssen Sie allerdings den vollständigen Pfad zum Programm angeben. Zum Abspielen einer MP3-Datei reicht `xmms` nicht aus, Sie müssen `/usr/bin/xmms` angeben.

Plugins

Plugins sind Erweiterungen mit Zusatzfunktionen für den Browser, beispielsweise zur Darstellung von Flash-Animationen, zum Abspielen von RealAudio-Dateien oder zur Ausführung von Java-Programmen. Wenn Sie in Mozilla die Adresse `about:plugins` eingeben, zeigt das Programm eine Liste aller installierten Plugins an. Das Ergebnis ist stark von der eingesetzten Distribution abhängig. Beachten Sie, dass gerade installierte Plugins erst nach

einem Neustart von Mozilla genutzt werden können! Einen Überblick über die verfügbaren Plugins sowie Tipps zu deren Installation finden Sie unter *http://plugindoc.mozdev.org/linux.ht*.

Leider gibt es eine Reihe von Plugins, die nur in Windows-Versionen zur Verfügung stehen und daher unter Linux nicht genutzt werden können. Abhilfe schafft das kommerzielle Programm CrossOver Plugin, mit dem Windows-Plugins auch unter Linux genutzt werden können (*http://www.codeweavers.com/*).

Mozilla sucht Plugin-Dateien im `plugins`-Verzeichnis relativ zum Mozilla-Installationsverzeichnis z. B. `/usr/lib/mozilla-n/plugins` bzw. `/opt/mozilla/lib/plugins/`, im Verzeichnis `~/.mozilla/plugins` (benutzerspezifische Plugin-Installationen) oder in den Verzeichnissen, die in der Umgebungsvariablen *MOZ_PLUGIN_PATH* angegeben sind.

Für Plugins ist übrigens keine eigene MIME-Konfiguration erforderlich. Die Plugin-Dateien enthalten alle erforderlichen Einstellungen.

Flash-Plugin: Zu den populärsten Plugins, die unter Linux zur Verfügung stehen, zählt Macromedia Flash zur Anzeige von Animationen. Download-Dateien in verschiedenen Linux-Paketformaten finden Sie unter `http://www.macromedia.com/shockwave/download`.

Bei der RPM-Variante erfolgt die Installation einfach durch `rpm -i paketname`. Die Plugin-Dateien landen im Verzeichnis `/usr/lib/flash-plugin`. Während der Installation wird automatisch ein Setup-Programm ausgeführt, das Links von Ihrem Mozilla-Plugin-Verzeichnis auf die neu installierten `*.so`- und `*.xpt`-Plugin-Dateien einrichtet.

Nach einem Browser-Neustart können Sie die Plugin-Installation beispielsweise mit der Seite *http://www.macromedia.com/shockwave/welcome/* testen.

Dort werden gleich zwei Macromedia-Plugins getestet, nämlich Shockwave- und Flash. Beachten Sie, dass das gerade installierte Flash-Plugin zwar den MIME-Type `application/x-shockwave-flash` (Flash) unterstützt, nicht aber `application/x-director` (Shockwave). Aus diesem Grund liefert die obige Seite eine Fehlermeldung (Shockwave-Plugin fehlt), selbst wenn das Flash-Plugin funktioniert. Für die Shockwave-Animationen würden Sie das Shockwave-Plugin benötigen, das zurzeit aber nur für Windows und MacOS verfügbar ist.

PDF-Plugin: Adobe Reader stellt ebenfalls ein Plugin für Mozilla zur Verfügung. Wenn Mozilla die Plugin-Datei findet, zeigt es PDF-Dateien direkt im Mozilla-Fenster an. Abhilfe ist einfach: Sobald Sie die Adobe-Reader-Plugin-Datei `nppdf.so` aus dem Mozilla-Plugin-Verzeichnis löschen, wird beim Anklicken von PDF-Dateien der Adobe-Reader in einem eigenen Fenster gestartet. Falls das nicht funktioniert, müssen Sie die MIME-Einstellungen im Mozilla-Konfigurationsdialog *Navigator* → *Hilfsanwendungen* anpassen: Dateien des Typs `application/pdf` sollen durch das Programm `acroread` dargestellt werden. Dabei müssen Sie den vollständigen Pfad des Programms angeben.

Java

Damit ein Java-Programm (ein so genanntes Applet) vom Webbrowser ausgeführt werden kann, muss ein Java-Interpreter zur Verfügung stehen. Die Verbindung zwischen dem Browser und dem Interpreter erfolgt in der Regel durch ein Plugin. Aus Lizenzgründen wird bei vielen Linux-Distributionen keine Java-Umgebung mitgeliefert. Im Internet finden Sie aber gleich *drei* verschiedene Java-Implementationen zum Download (Sun, Blackdown und IBM):

http://java.sun.com/j2se/downloads.html
http://www.ibm.com/developer/java/jdk/
http://www.blackdown.org

Wir beschränken uns auf die am weitesten verbreitete Sun-Variante. Von den vielen Dateien, die auf der Java-Website zur Auswahl stehen, benötigen Sie nur die J2SE (*Java 2 Standard Edition*). Davon gibt es aber wiederum zwei Varianten: Das JRE (*Java Runtime Environment*) reicht zur Ausführung von Java-Programmen aus. Wenn Sie dagegen selbst Java-Programme entwickeln möchten, benötigen Sie das SDK (*Software Development Kit*).

Mit dem Download erhalten Sie nicht direkt eine RPM-Datei, sondern ein Script. Dieses machen Sie mit `chmod u+x` ausführbar und starten es. Nachdem Sie die Lizenzbedingungen akzeptiert haben, erzeugt das Script die RPM-Datei, die Sie dann wie üblich mit `rpm -i paket.rpm` installieren. Die Installation erfolgt in das Verzeichnis `/usr/java/j2re-n`.

Anschließend müssen Sie nur noch Links vom Mozilla-Plugin-Verzeichnis zu den Java-Plugins erstellen. Das Plugin ist eigentlich für Netscape 6.1 gedacht, es ist aber zu den aktuellen Mozilla-Versionen kompatibel. Das Plugin steht für verschiedene Versionen des Compilers `gcc` zur Verfügung. Achten Sie darauf, dass Sie die richtige Version verwenden. (Aktuelle Versionen von Mozilla sind zumeist mindestens mit GCC 3.2 oder einer neueren Version kompiliert. Daher benötigen Sie das Plugin aus dem Verzeichnis `ns610-gcc32`.)

```
# cd /usr/lib/mozilla-n/plugins
# ln -s /usr/java/j2re1.4.2˙01/plugin/i386/ns610-gcc32/libjavaplugin˙oji.so .
```

Nachdem Sie die Installationsarbeiten abgeschlossen haben, wollen Sie natürlich testen, ob Java auch funktioniert. Dazu verwenden Sie am besten die JDK-Demoapplets, die Sie ausgehend von *http://java.sun.com/applets/* finden.

Denken Sie daran, dass Sie Java nicht nur installieren, sondern auch aktivieren müssen. Per Default ist das meist der Fall, aber wenn Java nicht funktioniert, lohnt auf jeden Fall ein Blick in den Mozilla-Konfigurationsdialog *Erweitert*!

5.1.2 Konqueror

`konqueror` ist integrativer Bestandteil des KDE-Desktops. Das Programm ist ein Universal-Browser (lokale Dateien, FTP, Web etc.). An dieser Stelle werden nur einige Besonderheiten bei der Verwendung als Webbrowser beschrieben.

Inhalt des Adressfelds löschen: Der Button in Form eines schwarzen X in der Adressleiste löscht deren Inhalt. Das klingt banal, ist in Wirklichkeit aber ausgesprochen praktisch: Sie können eine Webadresse mit der Maus markieren, per Mausklick auf das X die bisherige Adresse im Adressfeld löschen und dann per Klick mit der mittleren Maustaste den markierten Text einfügen. (Bei den meisten anderen Webbrowsern ist der bisherige Inhalt im Adressfeld dagegen im Weg und muss erst gelöscht werden. Dabei geht dann oft der Inhalt der X-Ad-hoc-Zwischenablage verloren.)

Popup-Fenster abstellen: Sie finden im Konfigurationsdialog *JavaScript* die Option, die das Erzeugen neuer Fenster deaktiviert bzw. mit einer Rückfrage verbindet.

Übersetzung von Webseiten: Ein nettes Feature ist das Kommando *Extras → Webseite übersetzen*. `konqueror` führt die Übersetzung (z. B. Englisch → Deutsch) natürlich nicht selbst durch, sondern leitet die Seite an „Babelfish“ weiter. Das Ergebnis ist zwar meist erheiternd, aber man kann zumindest erraten, worum es auf der Webseite geht.

Schnellsuche: Für einzelne Websites (z. B. für Suchmaschinen) können Abkürzungen definiert werden. Wenn Sie beispielsweise als Adresse `gg:abc` eingeben, wird bei *http://www.google.com* eine Suche nach dem Begriff `abc` durchgeführt. Im Konfigurationsdialog *Web-Tastenkürzel* können Sie die Abkürzungsliste durch eigene Kürzel ergänzen.

Browser-Identifizierung: Manche Websites unterstützen nur den Internet Explorer und eventuell noch Mozilla/Netscape. Wenn Sie Konqueror als Browser einsetzen, sehen Sie statt der gewünschten Seite meist nur die Aufforderung, einen anderen Browser zu verwenden. Geben Sie einfach im Konfigurationsdialog *Browser-Identifizierung* für die betreffende Website an, dass Konqueror eine andere Identifikationszeichenkette an den Webserver senden soll.

Textgröße bei der Darstellung von Textdateien: Generell können Sie die gewünschte Textgröße im Konfigurationsdialog einstellen. Gelegentlich verweisen Links von Webseiten aber nicht auf andere HTML-Dokumente, sondern auf einfache Textdateien (ASCII-Dateien). Derartige Dokumente werden nicht direkt von `konqueror` dargestellt, sondern von `kwrite`. Deswegen werden hierfür die `konqueror`-Font-Einstellungen ignoriert. Wenn Sie also möchten, dass Textdateien in einer größeren Schrift angezeigt werden, starten Sie `kwrite` und verändern dessen Font-Einstellungen.

Webseite als Archiv speichern: Mit *Extras → Archive Web Page* können Sie alle Dateien einer Webseite in einem komprimierten Archiv speichern (d. h. alle Frame-Seiten, alle Bilder etc. – einfach alles, was notwendig ist, um die Seite

exakt so darzustellen, wie dies momentan der Fall ist). Die Archivdatei erhält die Kennung `*.war`, es handelt sich aber um ein einfaches `tar`-Archiv, das mit `gzip` komprimiert ist.

Plugins: Konqueror nutzt in der Regel dieselben Plugins wie Mozilla. Wenn die Plugins nicht funktionieren, werfen Sie einen Blick in den Konfigurationsdialog *Plugins*: Dort können Sie nachsehen, welche Plugins Konqueror gefunden hat, und einstellen, welche Verzeichnisse nach Plugins durchsucht werden. Mit dem Button *Nach neuen Plugins suchen* können Sie neu installierte Plugins in die Konqueror-Plugin-Liste aufnehmen.

Java: Beim Konqueror erfolgt die Kommunikation mit der Java VM nicht durch ein Plugin, sondern über den KJAS (KDE Java Applet Server). KJAS wiederum startet direkt den Java-Interpreter, also das Programm `java`. Falls Konqueror Probleme hat, diese Datei zu finden, können Sie den vollständigen Dateinamen mit `which java` ermitteln und dann im Konqueror-Konfigurationsdialog *Java und JavaScript* eintragen.

5.1.3 Lynx

Lynx hat gegenüber anderen WWW-Browsern einen großen Vorteil: Es läuft im Textmodus, wobei viele Merkmale des WWW natürlich wegfallen (Textattribute, Bilder, Animationen). Dafür ist Lynx aber sehr genügsam, was den Verbrauch von Speicher und Rechenkapazität angeht, und es setzt keine X-Installation voraus. Das Programm ist auch ideal dazu geeignet, rasch Online-Dokumentation im HTML-Format zu lesen oder in Text zu konvertieren.

Sie starten das Programm im Regelfall dadurch, dass Sie eine WWW-Adresse oder den Namen einer HTML-Datei als Parameter angeben. Lynx lädt das Dokument und zeigt die erste Seite an. Lynx kann durch die Angabe eines Verzeichnisses gestartet werden: Das Programm zeigt dann eine Liste aller darin enthaltenen Dateien und Unterverzeichnisse an, die wie Querverweise ausgewählt werden können. Textdateien im ASCII-Format können direkt in Lynx angezeigt werden. Neuere Versionen von Lynx zeigen Überschriften und Links in unterschiedlichen Farben an und können per Maus bedient werden. (Mit der linken Taste folgen Sie einem Link, die rechte Taste führt zur vorherigen Seite zurück.) Diese Funktionen können mit den Optionen `-color` und `-use_mouse` aktiviert werden.

Lynx hat noch eine weitere Funktion: Das Programm kann als Konverter HTML → ASCII verwendet werden und so HTML-Dokumente in eine gut lesbare Textdatei verwandeln. Das folgende Kommando demonstriert diese Anwendung:

```
lynx -dump quelle.html > ziel.txt
```

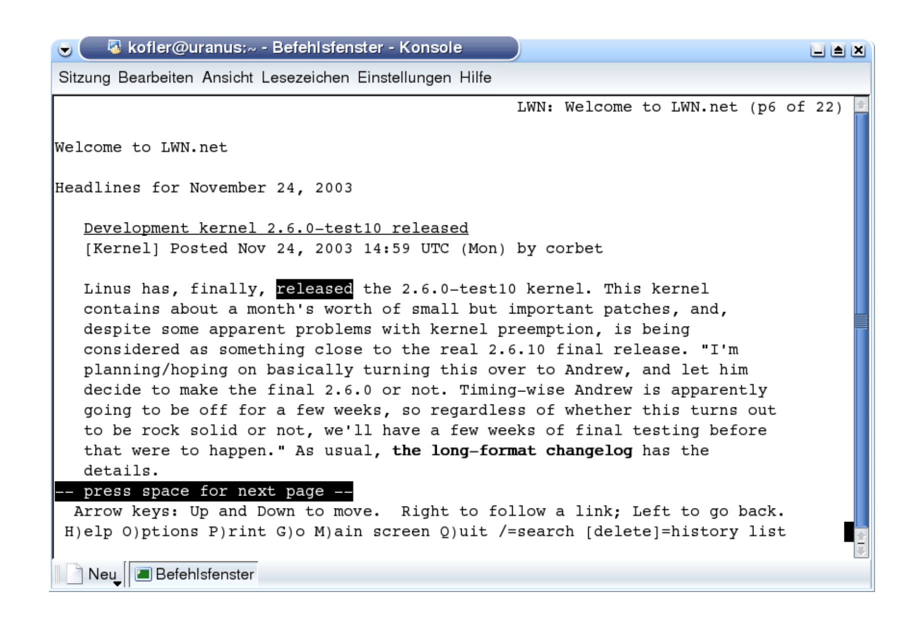


Abbildung 5.1: Lynx – HTML-Dateien im Textmodus lesen

5.2 E-Mail

Grundsätzlich gehen wir in diesem Abschnitt davon aus, dass Sie die E-Mails lokal auf Ihrem Rechner mit einem E-Mail-Programm (E-Mail-Client) bearbeiten möchten. Die Alternative dazu wäre die Nutzung eines der vielen Web-Mail-Systeme, wie Sie von GMX, Yahoo etc. angeboten werden. Dazu benötigen Sie kein eigenes E-Mail-Programm, sondern nur einen Webbrowser.

5.2.1 Glossar

Account: Wenn Ihnen ein Internet-Provider oder eine Firma eine E-Mail-Adresse zur Verfügung stellt, wird dieser Service auch als Account (Konto) bezeichnet. Da viele Leute mehrere E-Mail-Adressen besitzen, können die meisten E-Mail-Programme mehrere Accounts verwalten.

Attachment (Anlage): Zusammen mit Ihrer Nachricht können Sie so genannte Attachments (Anlagen) versenden. Dabei handelt es sich einfach um Dateien, die zur E-Mail hinzugefügt werden (z. B. Bilder). Sie sollten allerdings nie E-Mails versenden, die nur das Attachment enthalten. Geben Sie immer auch einen Text an, der den Inhalt des Attachments kurz beschreibt.

Kopfzeilen: Am Beginn einer jeden E-Mail stehen einige Kopfzeilen, die die Adresse, den Betreff und eventuell einige weitere Informationen enthalten.

E-Mail-Kopfzeilen

`From: adresse`	E-Mail-Adresse des Absenders
`To: adresse`	E-Mail-Adresse des Empfängers
`Subject: inhalt`	Kurzbeschreibung des Inhalts (eine Zeile)
`Cc: adresse`	Kopie an die Cc-Adresse senden (*carbon copy*)
`Bcc: adresse`	Wie Cc:, allerdings erfährt der eigentliche Adressat nicht, dass die Mail auch an eine andere Person versandt wurde (*blind carbon copy*).
`Reply-To: adresse`	Antwort nicht an `From:`, sondern an die hier angegebene Reply-Adresse senden

Signatur: Oft sollen alle E-Mails mit einigen Zeilen Text abgeschlossen werden, die etwa den Firmennamen, eine Adresse etc. enthalten können. Damit Sie diesen Text nicht ständig neu eingeben müssen, können Sie bei den meisten E-Mail-Programmen eine so genannte Signatur definieren. Manche verwenden dazu automatisch die Datei `~/.signature`.

MIME: Der Begriff MIME (*Multipurpose Internet Mail Extensions*) wird bei E-Mail-Programmen auf zwei verschiedene Weisen verwendet. Einerseits hilft die richtige MIME-Konfiguration dem E-Mail-Programm, richtig mit Attachments umzugehen und eine per E-Mail versandte MP3-Datei mit einem MP3-Player zu öffnen. Andererseits kann sich MIME auch auf die Kodierung des Nachrichtentexts beziehen. Ursprünglich war E-Mail nur für 7-Bit-ASCII-Zeichen gedacht. Nachrichten mit Sonderzeichen (also mit Zeichen, deren Code größer als 127 ist) bereiteten Schwierigkeiten. Um diese Probleme zu umgehen, wurden verschiedene MIME-Typen zur Kodierung der Nachrichten definiert.

POP: Zur Übertragung von E-Mails vom Provider auf Ihren Rechner kommt üblicherweise das *Post Office Protocol* (POP) zum Einsatz. Damit das E-Mail-Programm mit dem Provider kommunizieren kann, benötigt es drei Informationen: die Adresse des POP-Servers, den POP-Login-Namen und das POP-Passwort.

IMAP: Eine Alternative zu POP ist das *Internet Message Access Protocol*. Der Hauptunterschied zu POP besteht darin, dass bei IMAP die E-Mails üblicherweise auf dem IMAP-Server bleiben. Das E-Mail-Programm dient in diesem Fall also nur zur Kommunikation mit dem Server. IMAP ist dann optimal, wenn Sie Ihre E-Mails von unterschiedlichen Rechnern aus bearbeiten möchten, ohne die Nachteile eines Web-Mail-Systems in Kauf zu nehmen.

SMTP: Zum Versenden eigener E-Mails wird das *Simple Mail Transfer Protocol* (SMTP) verwendet. Zur Kommunikation mit dem SMTP-Server des Providers benötigt das E-Mail-Programm meist nur die Adresse des SMTP-Servers. Ob auch beim SMTP-Server eine Authentifizierung erforderlich ist, hängt vom Provider ab. Bei vielen Providern gilt eine vorherige POP-Authentifizierung automatisch auch für SMTP. Andere Provider verlangen eine eigene SMTP-Authentifizierung.

Hinweis Während der Account-Konfiguration können Sie bei vielen E-Mail-Programmen die Port-Nummer für POP, IMAP und SMTP angeben. Das ist aber selten erforderlich, weil die Default-Werte zumeist zutreffen (POP = 110, IMAP = 143, SMTP = 25).

5.2.2 Mail-Konfiguration

Es gibt zwei Möglichkeiten zur Kommunikation zwischen dem E-Mail-Programm und dem Internet-Provider (der Ihnen eine E-Mail-Adresse zur Verfügung stellt). Die einfache Variante besteht darin, dass sich das E-Mail-Programm um alle Aufgaben selbst kümmert (Standalone-Lösung). Die andere Variante entspricht dem klassischen Unix-Ansatz, wonach Teilaufgaben nach Möglichkeit auf kleine, überschaubare Programme verteilt werden: Ein Programm ist für das Versenden von E-Mails zuständig, ein Programm für das Empfangen von E-Mails und das dritte Programm für deren Bearbeitung. Die Konfiguration ist nun zwar ein bisschen komplizierter, sie hat aber in größeren Netzwerken auch manche Vorteile.

Die Ausgangsdaten sind in jedem Fall dieselben. Ihr E-Mail-Provider wird Ihnen Daten zur Verfügung stellen, die etwa wie die folgenden Zeilen aussehen. (Wenn Sie sich in einem lokalen Netzwerk befinden, kann durchaus auch ein interner Server innerhalb des lokalen Netzes als E-Mail-Provider dienen!)

```
E-Mail-Adresse:              user@provider.de
SMTP-Mail-Server:            mail.provider.de
POP- oder IMAP-Mail-Server:  pop.provider.de
Login-Name (Benutzername):   user02
Passwort:                    qwe44trE
```

Je nach Provider fehlt der Login-Name bzw. Benutzername – dann müssen Sie stattdessen die E-Mail-Adresse oder den Beginn der E-Mail-Adresse (im Beispiel „user“) oder den Login-Namen für den Internet-Zugang angeben. Es kann auch sein, dass Sie zwei Login/Passwort-Paare bekommen: eines für POP und eines für SMTP.

Wenn Sie Ihre E-Mail-Konfiguration ausprobieren möchten, senden Sie sich einfach selbst eine E-Mail. Wenn es gelingt, die E-Mail zu versenden und anschließend wieder zu empfangen, funktioniert alles. Überprüfen Sie auch, dass die Absenderadresse korrekt ist. (Probieren Sie aus, ob bei *Reply-to* tatsächlich Ihre E-Mail-Adresse verwendet wird.) Alle E-Mail-Clients bieten die Möglichkeit an, die abgeholten E-Mails auf dem Server zu belassen. Das ist zwar in der Regel keine gute Idee (weil der Platz dort meist beschränkt ist); während der Testphase stellt diese Option aber sicher, dass keine E-Mails verloren gehen können.

Standalone-Konfiguration: Wenn Sie unter Windows schon einmal einen E-Mail-Zugang konfiguriert haben, wird Ihnen diese Variante bekannt vorkommen: *Ein* Programm (z. B. Netscape oder Outlook) wird für alle E-Mail-Aufgaben

genutzt: für das Lesen und Verfassen von Nachrichten, für deren Speicherung bzw. Verwaltung sowie für den Empfang und Versand. Das E-Mail-Programm arbeitet also gleichsam als Standalone-System. Auch unter Linux unterstützen alle neueren E-Mail-Clients mit der Ausnahme einiger textbasierter Programme dieses Konzept. Sie müssen im E-Mail-Programm bei der Account-Konfiguration einfach die oben genannten Daten angeben.

Unix-typische Konfiguration: Ein wesentlicher Nachteil der Standalone-Lösung besteht in ihrer Redundanz: Jeder E-Mail-Client muss Programmteile für die POP- und SMTP-Kommunikation enthalten. Unter Unix/Linux ist es eigentlich üblich, derartige Aufgaben dezentral zu verteilen. Deswegen sieht ein herkömmliches Unix-E-Mail-System ganz anders aus:

- **E-Mails beim Provider abholen (fetchmail):** Zum Abholen von E-Mails wird üblicherweise das Programm `fetchmail` eingesetzt. Es holt neu eingetroffene E-Mails mithilfe von POP oder IMAP beim Provider ab und speichert sie in einer so genannten Inbox-Datei, die üblicherweise den Dateinamen `/var/spool/mail/benutzername` hat. Textbasierte E-Mail-Programme wie `mutt` oder `pine` lesen E-Mails per Default aus dieser Datei. Zur `fetchmail`-Konfiguration sind dieselben Angaben wie bei der POP-Konfiguration eines Standalone-Systems erforderlich. `fetchmail` kann z. B. durch ein `cron`-Script regelmäßig ausgeführt werden.
- **E-Mails versenden:** Zum Versenden von E-Mails wird ein so genannter *Mail Transport Agent* (MTA) eingesetzt. Je nach Distribution kommt hierfür meist `sendmail`, `postfix` oder `qmail` zum Einsatz. In jedem Fall läuft der MTA als Systemdienst (Dämon). Dieser kümmert sich darum, dass die E-Mails an die richtigen Adressen weitergeleitet werden. Sind E-Mails nicht an den lokalen Rechner adressiert, nimmt `sendmail` Kontakt zu anderen E-Mail-Servern auf (Protokoll SMTP). Wenn gerade keine Verbindung zu anderen E-Mail-Servern möglich ist, werden die E-Mails zwischengespeichert. Auf den meisten Linux-Rechnern läuft ein MTA allein deswegen, um auf dem lokalen Rechner für alle Programme und Benutzer E-Mail-Dienste zur Verfügung zu stellen. Selbst wenn Sie den Rechner allein nutzen, ist das sehr praktisch: Wenn beispielsweise der Ausdruck einer Datei nicht gelingt, versendet der Druckdämon eine E-Mail, um auf das Problem hinzuweisen. Aber das klappt natürlich nur, wenn ein lokales E-Mail-System eingerichtet ist.

Das E-Mail-Programm kann sich bei diesem Szenario auf seine Kernaufgaben konzentrieren – das Lesen, Speichern und Verfassen von E-Mails. Es benötigt keine POP-Funktionen, sondern kann die E-Mails einfach aus einer Datei lesen. Ebensowenig benötigt es SMTP-Funktionen, sondern kann zu versendende E-Mails einfach an den MTA übergeben. Leider gibt es einige neuere E-Mail-Clients wie Mozilla, die das klassische Unix-System nicht unterstützen und darauf bestehen, die SMTP- und POP-Kommunikation selbst durchzuführen.

5.2.3 Mailbox-Formate und -Konvertierung

mbox-Format: Alle E-Mail-Programme bieten die Möglichkeit, eingetroffene oder selbst verfasste E-Mails in Verzeichnissen zu speichern. Intern speichern die meisten E-Mail-Programme die E-Mails eines Verzeichnisses in einer Mailbox-Datei (mbox-Format). Die E-Mails werden einfach zu einer langen Textdatei verbunden. Zur Trennung zwischen den E-Mails dient mindestens eine Leerzeile, gefolgt von einer Zeile, die mit „From" beginnt.

Die meisten E-Mail-Clients erzeugen neben den mbox-Dateien zusätzliche Verwaltungsdateien mit einem Inhaltsverzeichnis zu den Mailbox-Dateien. Diese Verwaltungsdateien beschleunigen den Zugriff auf einzelne E-Mails, sind aber nicht zwischen den E-Mail-Programmen kompatibel. Auch die Art und Weise, ob und wie Hierarchien von Mail-Verzeichnissen unterstützt werden, hängt vom jeweiligen E-Mail-Programm ab. Das mbox-Format erleichtert den Wechsel des E-Mail-Clients. Normalerweise müssen Sie lediglich die mbox-Dateien (ohne die Verwaltungsdateien) in das Verzeichnis kopieren, in dem der E-Mail-Client sie erwartet.

maildir-Format: Neben dem mbox-Format unterstützen manche E-Mail-Programme (z. B. KMail) auch das maildir-Format. Dabei wird jede einzelne E-Mail in einer eigenen Datei gespeichert. Eine Mailbox besteht aus allen Dateien innerhalb eines Verzeichnisses. Der offensichtliche Vorteil besteht darin, dass einzelne Nachrichten einfacher gelöscht werden können.

Konversion von Outlook Express-E-Mails: Unter Windows gibt es keinen gemeinsamen Standard zur Speicherung von E-Mails. Zwar gibt es auch Windows-Programme, die das mbox-Format verwenden; die Mehrheit der E-Mail-Clients verwendet aber ihr eigenes Format. Um Ihre E-Mails von einem Windows- auf einen Linux-Rechner zu überspielen, müssen Sie Ihre E-Mails in das mbox-Format konvertieren. Zu diesem Zweck gibt es glücklicherweise schon eine ganze Reihe von Programmen (suchen Sie einfach im Web nach *mbox conversion outlook eudora*). Generell ist es eine gute Idee, vor dem Start irgendeines Konverters alle E-Mail-Verzeichnisse zu komprimieren. Dabei werden alle gelöschten E-Mails endgültig aus den Mailbox-Dateien entfernt. Führen Sie in Outlook Express das Kommando *Datei → Ordner → Alle Ordner komprimieren* aus!

OE5/6 Multi Converter: Dieses kostenlose Programm läuft unter Windows. Es konvertiert OE-E-Mails in drei verschiedene Formate, darunter in das mbox-Format. Die Anwendung ist unkompliziert: Sie müssen lediglich das OE-Konto auswählen, das gewünschte Zielformat sowie ein Zielverzeichnis angeben. Leider geht dabei die OE-Verzeichnisstruktur bei der Konvertierung verloren, und Verzeichnisse mit News-Beiträgen werden wie E-Mail-Verzeichnisse behandelt (*http://www.tietew.net/soft/OE2/index.en.html*).

Import mit Mozilla für Windows durchführen: Im Gegensatz zur Linux-Version besitzt die Windows-Version von Mozilla eine Funktion zum E-Mail-Import von Eudora, Outlook und Outlook Express. Bei einem Outlook-Express-

Import landen die importierten E-Mails im mbox-Format im folgenden Verzeichnis:

```
C:\Dokumente und Einstellungen\Benutzername\Anwendungsdaten\
   Mozilla\Profiles\Default\xxxxxx.xxx\Mail\Local Folders\
```

Das Verzeichnis wurde hier nur aus Platzgründen auf mehrere Zeilen verteilt. `xxxxx.xxx` ist der zufällige Name eines Verzeichnisses, das Mozilla beim Anlegen des Accounts erzeugt. Mozilla speichert jede Mailbox in der Datei `name`, zu der sich die Indexdatei `name.msf` gesellen kann. Ordner werden durch Verzeichnisse mit dem Namen `name.sbd` abgebildet.

Die Mailbox-Dateien `name` können Sie nun direkt in das Mail-Verzeichnis eines beliebigen Linux-E-Mail-Programms kopieren. Wenn Sie auch unter Linux mit Mozilla arbeiten, ist es noch besser, alle konvertierten Dateien und Verzeichnisse in das äquivalente `Local Folders`-Verzeichnis zu kopieren. Der Vorteil dieser Vorgehensweise besteht darin, dass dabei auch die Ordnerstruktur von Outlook Express erhalten bleibt.

```
user$  cd ~/.mozilla/default/xxxxxx.xxx/Mail/Local Folders/
user$  cp -a "windowsverzeichnis/Outlook Express Mail*" .
```

Adressbuch: Für die Speicherung des Adressbuchs gibt es leider weder unter Windows noch unter Linux einen gemeinsamen Standard. Auch hier gilt aber, dass es im Internet eine Menge Konversionsprogramme gibt. In vielen Fällen gelingt aber auch ein Export/Import in Form einer CSV-Textdatei (*comma separeted values*).

5.3 E-Mail-Clients

Unter Unix/Linux werden die E-Mail-Programme oft auch als „E-Mail-Clients“ oder „Mail User Agents“ (MUA) bezeichnet, um sie von anderen Programmen zur E-Mail-Bearbeitung zu unterscheiden, z. B. von *Mail Transport Agents* wie `sendmail` oder `postfix`. Neben den hier behandelten Programmen gibt es viele andere, auf die wir aus Platzgründen nicht eingehen. Es seien hier nur noch die Kommandos `mail` und `mailx` erwähnt, die nicht für die interaktive Verwendung gedacht sind, sondern um E-Mails aus einem Shell- oder Perl-Script heraus zu versenden. `cone`, `mutt` und `pine` sind jeweils sehr leistungsfähige Clients, die allerdings nur im Textmodus laufen.

5.3.1 Mozilla und Thunderbird

Vom Mozilla-Webbrowser gelangen Sie mit *Fenster* → *Mail* bzw. mit (Strg)+(2) in den Mail- und News-Client. Wenn dieses Kommando nicht zur Verfügung steht, müssen Sie das Mozilla-Mail-Paket installieren. (Bei manchen Distributionen wird per Default oft nur der Browser installiert.) Der neue E-Mail-Client Mozilla

Thunderbird ist dem hier beschriebenen klassischen Mozilla-Mail-Client sehr ähnlich, so dass wir nur auf Mozilla eingehen.

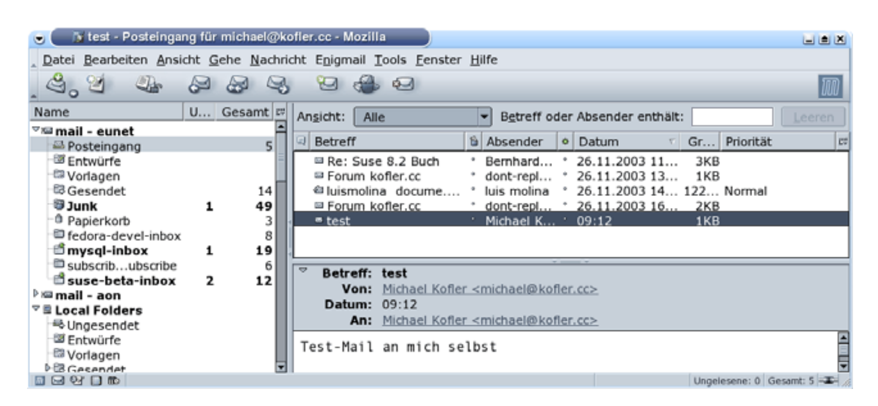


Abbildung 5.2: Der Mozilla-Mail-Client

Konfigurationsdialoge: Die Konfigurationseinstellungen sind über zwei Dialoge verteilt: Globale Optionen der E-Mail-Komponente von Mozilla werden im Mozilla-Konfigurationsdialog eingestellt – Menükommando *Bearbeiten* → *Einstellungen*. Für Account-spezifische Optionen gibt es dagegen einen zweiten Dialog, der mit *Bearbeiten* → *Mail-Account-Einstellungen* geöffnet wird. Es ist leider nicht immer klar nachvollziehbar, welche Option in welchem Dialog zu finden ist. Daher wird auf den folgenden Seiten immer angegeben, welche Einstellung im Mozilla- bzw. im Account-Konfigurationsdialog erfolgt.

Account einrichten: *Datei* → *Neu* → *Account* führt zu einem Assistenten, der Ihnen beim Einrichten des neuen Accounts hilft. Im ersten Schritt geben Sie Ihren tatsächlichen Namen und Ihre E-Mail-Adresse an. Als Nächstes müssen Sie sich zwischen POP bzw. IMAP entscheiden und die Adresse des POP- oder IMAP-Servers angeben. Beim ersten Account müssen Sie auch die Adresse des SMTP-Servers zum Versenden Ihrer E-Mails angeben.

Im nächsten Schritt geben Sie den Login-Namen für den POP- bzw. IMAP-Server an. (Der Konfigurationsdialog bezeichnet diesen Namen als *Benutzername*.) Zuletzt geben Sie dem neuen Account einen Namen. Wenn Sie vorhaben, mehrere E-Mail-Accounts einzurichten, ist es sinnvoll, den Account-Namen aus der E-Mail-Adresse und dem Namen des Providers zusammenzusetzen.

Beim ersten Verbindungsversuch fragt das Programm nach dem Passwort für den POP- bzw. IMAP-Login. Wenn Sie die entsprechende Option anklicken, wird das Passwort von Mozilla gespeichert (siehe auch den Mozilla-Konfigurationsdialog *Datenschutz* → *Passwörter*).

Nachträgliche Änderungen können Sie im Account-Konfigurationsdialog in den Dialogblättern *Server-Einstellungen* bzw. *Server für ausgehende Nachrichten* durchführen. Beachten Sie, dass Mozilla zwar beliebig viele E-Mail-Accounts

zulässt, aber nur die Konfiguration eines SMTP-Servers zum Versenden von E-Mails vorsieht. Das muss auch immer der SMTP-Server Ihres Providers sein (auch wenn Sie Ihre E-Mails von einem anderen Rechner holen).

Per Default werden zur SMTP-Authentifizierung dieselben Informationen (Login-Name, Passwort) wie für POP bzw. IMAP verwendet. Viele E-Mail-Provider ignorieren diese Informationen aber. E-Mails dürfen jedoch nur dann versandt werden, wenn vorher eine POP-Authentifizierung stattgefunden hat. Wenn Ihr Provider andere Authentifizierungswünsche hat, können Sie die notwendigen Einstellungen im Dialogblatt *Server für ausgehende Nachrichten* vornehmen.

Neben dem bzw. den Accounts für Ihre E-Mail-Provider existiert unter Mozilla immer noch ein weiterer Account: *Local Folders* bzw. *Lokale Ordner*. Dieser Account ist zur Archivierung von E-Mails gedacht. Alle darin gespeicherten E-Mails werden auf der lokalen Festplatte gespeichert und sind dort immer verfügbar. Es ist aber unmöglich, E-Mails in diesem Spezial-Account zu empfangen.

Tipp

Wenn Sie sich während der Definition eines E-Mail-Accounts beim Login-Namen bzw. beim Passwort vertippen, zeigt Mozilla möglicherweise alle paar Sekunden eine Warnung an, dass die Authentifizierung nicht funktioniert hat – ohne aber die Möglichkeit einer Korrektur anzubieten! Ein sinnvolles Arbeiten bzw. eine Änderung der Account-Einstellungen ist wegen der ständigen Warnungen unmöglich. Abhilfe: Schließen Sie das E-Mail-Fenster, öffnen Sie aus dem Mozilla-Hauptfenster heraus den Mozilla-Konfigurationsdialog *Datenschutz* → *Passwörter* und löschen Sie das Passwort für den Mailbox-Account (Button *Gespeicherte Passwörter verwalten*).

Unix-typische Account-Konfiguration: Die Mozilla-Account-Konfiguration sieht nicht vor, E-Mails direkt an einen MTA wie `sendmail` zu übergeben. Beachten Sie, dass auf den meisten Linux-Rechnern per Default ein SMTP-Server läuft, um lokale E-Mails zu versenden. Geben Sie als *Servername* „localhost" an. Ebenso werden Sie vergebens nach einer Möglichkeit suchen, E-Mails aus einer lokalen Inbox-Datei zu lesen, also z. B. aus `/var/spool/mail/username`. Auch das ist möglich, Sie müssen aber die Konfigurationsdatei `prefs.js` manuell ändern. Details dazu finden Sie in der deutschen Mozilla-FAQ (*http://www.holgermetzger.de/faq.html*).

Mailbox-Interna: Mozilla erzeugt für jeden Account ein eigenes Verzeichnis. Diese Verzeichnisse befinden sich im Mozilla-Mail-Verzeichnis mit dem Namen `~/.mozilla/default/xxxxxxxx.xxx/Mail/`. Dabei ist `xxxxx.xxx` eine zufällige Zeichenkette, die Mozilla beim ersten Start erzeugt. Innerhalb der Account-Verzeichnisse wird jede Mailbox in der Datei `name` gespeichert (mbox-Format). Parallel dazu erzeugt Mozilla eine Indexdatei `name.msf`. Mailbox-Unterverzeichnisse haben den Namen `name.sbd/`.

E-Mail-Formatierung (HTML-Mail): Beim Verfassen neuer E-Mails können Sie verschiedene Schriften, Farben etc. einsetzen. per Default wird die E-Mail dann sowohl als HTML-Dokument als auch als reine Textnachricht versandt, so dass jedes gängige E-Mail-Programm die E-Mail korrekt darstellen kann. Wenn Sie sinnvollerweise das HTML-Zeug abstellen wollen, gibt es mehrere Möglichkeiten:

- Beim Verfassen einer neuen E-Mail können Sie mit *Optionen* → *Format* das gewünschte Format (Text, HTML oder beides) auswählen.
- Im Mozilla-Konfigurationsdialog *Mail* → *Sendeformat* können Sie Regeln aufstellen, welche Adress-Domains nur reine Text-Mails und welche Domains HTML-Mails akzeptieren. Mozilla passt neue E-Mails dann automatisch an das erforderliche Format an.
- Wenn Sie – wie viele fortgeschrittene E-Mail-Anwender – generell auf die HTML-Formatierung von E-Mails verzichten möchten, deaktivieren Sie im Account-Konfigurationsdialog die Option *Nachrichten im HTML-Format verfassen*.

Im Mozilla-Konfigurationsdialog *Mail* → *Nachrichtenansicht* können Sie angeben, wie empfangene reine Text-E-Mails dargestellt werden sollen: mit einer Schriftart in fester oder variabler Breite, mit oder ohne Textumbruch etc.

Versandte E-Mails nochmals senden: Manchmal kommt es vor, dass der Empfänger eine E-Mail nicht erhält (*mailbox full*). Zumeist möchten Sie die E-Mail dann nach ein oder zwei Tagen nochmals versenden. Dazu markieren Sie die E-Mail im Ordner *Gesendet* und führen *Nachricht* → *Nachricht als neu bearbeiten* aus (kurz (Strg)+(E)).

Thread-Ansicht: Bei E-Mails aus Mail-Listen kann durch die Thread-Ansicht die Struktur des Nachrichtenverkehrs nachvollzogen werden. Diese Ansicht aktivieren Sie, indem Sie das kleine Thread-Icon am linken Rand der E-Mail-Liste anklicken (links neben dem *Betreff*-Feld).

Signaturen: Im Account-Konfigurationsdialog können Sie eine Datei auswählen, die Ihre Signatur enthält. Der Text wird automatisch an das Ende einer jeden neuen E-Mail angefügt. Beachten Sie, dass eine Signatur nicht mehr als vier bis fünf Zeilen lang sein sollte.

Filter für den Posteingang: Wenn Sie viele E-Mails aus Mailing-Listen erhalten, kann es zweckmäßig sein, eintreffende E-Mails automatisch auf verschiedene Verzeichnisse zu verteilen. Solche Filter können Sie mit *Tools* → *Nachrichtenfilter* einrichten.

Spam-Filter: Mozilla verwendet per Default einen Spam-Filter, der versucht, unerwünschte E-Mails zu erkennen und als Junk-Mail zu identifizieren. Die Trefferquote ist überraschenderweise von Anfang an relativ hoch, kann aber noch

verbessert werden: Dazu müssen Sie eintreffende E-Mails, die falsch kategorisiert werden, mit *Nachricht → Markieren → Als (kein) Junk* explizit als Spam bzw. als korrekte E-Mail identifizieren.

Nach einigen Tagen bzw. nach einigen Hundert Spams ist die Trefferquote so hoch, dass Sie den Spam-Filter mit *Tools → Junk-Mail-Filter* so konfigurieren können, dass Spams automatisch in ein Junk-Verzeichnis umgeleitet werden. Allerdings müssen Sie hin und wieder einen Blick in dieses Verzeichnis werfen, weil es gelegentlich (jedoch selten) passiert, dass E-Mails irrtümlich als Spam identifiziert werden.

Externe Grafiken unterdrücken: Spam-Mails enthalten oft Links auf Grafikdateien, die sich im Internet befinden und somit nicht Teil der E-Mail sind. Mozilla zeigt derartige Grafiken normalerweise an. Damit kann der Versender der E-Mail erkennen, dass Sie die E-Mail erhalten und gesehen haben. Aus diesem Grund kann es wünschenswert sein, die Anzeige solcher Grafiken zu unterbinden. Aktivieren Sie einfach die Option *Keine externen Grafiken in Mail-Nachrichten laden* im Mozilla-Konfigurationsdialog *Datenschutz → Grafiken*.

Enigmail: Enigmail ist eine relativ neue Komponente von Mozilla, die das automatische Signieren bzw. Verschlüsseln von E-Mails ermöglicht. Das setzt allerdings voraus, dass Sie sich vorher ein eigenes Zertifikat besorgen. Dieser Vorgang und die nachfolgende Konfiguration von Enigmail ist in der Hilfe ausführlich beschrieben. Wenn Sie Enigmail nicht nutzen möchten, wählen Sie im Mozilla-Konfigurationsdialog *Datenschutz → Enigmail* die Option *No default encryption*.

5.3.2 KMail

KMail (*http://kmail.kde.org/*) ist der E-Mail-Client des KDE-Desktop-Projekts. Das Programm ist außergewöhnlich leistungsstark, sehr schnell und bietet noch mehr Konfigurationsoptionen als Mozilla.

Account-Konfiguration: Gehen Sie mit *Einstellungen → KMail einrichten* in den KMail-Konfigurationsdialog. per Default ist bereits eine so genannte Identität *Standard* eingerichtet. Mit *Ändern* stellen Sie hier Ihren tatsächlichen Namen, die E-Mail-Adresse und bei Bedarf weitere Optionen – z. B. die Signatur – ein.

Die eigentliche Account-Konfiguration erfolgt im Dialogblatt *Netzwerk*. Sie müssen zumindest ein Ausgangs- und ein Eingangspostfach einrichten. Mit dem Ausgangspostfach ist im Regelfall die Konfiguration des SMTP-Servers gemeint. Als Alternative können Sie auch die Versandart *Sendmail* wählen – dann übergibt KMail die zu versendenden Nachrichten direkt an den lokalen Mail-Server (egal, ob es sich dabei tatsächlich um `sendmail`, `postfix` oder `qmail` handelt). Wenn Ihr E-Mail-Provider keine SMTP-Authentifizierung vorsieht und stattdessen eine vorherige POP-Authentifizierung verlangt, müssen Sie die Option *Nachrichten bei Überprüfung der Postausgangsfächer senden* aktivieren.

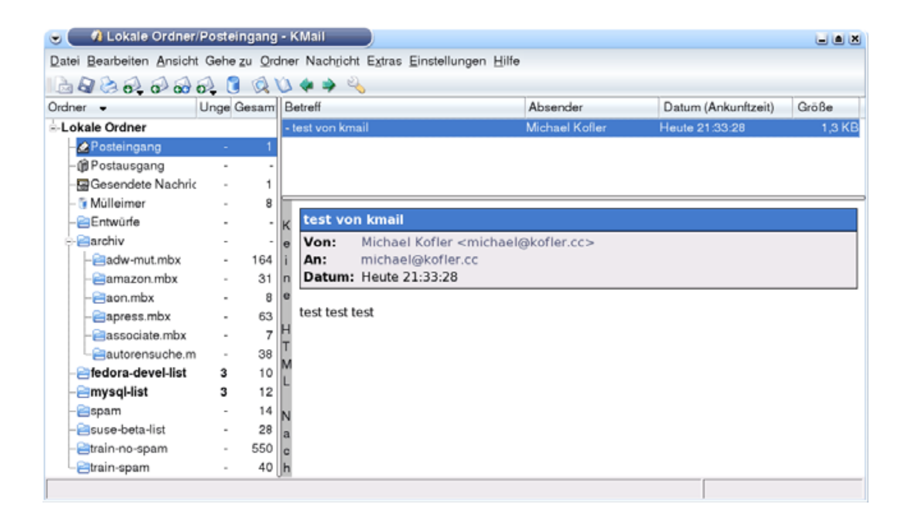


Abbildung 5.3: Das Mail-Programm KMail

Das Eingangspostfach ist zumeist ein POP-Server. Hier geben Sie den Benutzernamen (POP-Login-Namen), das Passwort und den POP-Server-Namen an. Anschließend sollten Sie mit *Extras* → *Fähigkeiten des Servers testen* KMail die optimale Detaileinstellung überlassen. (Je nach Provider kann es passieren, dass KMail das Server-Zertifikat beanstandet. Im Regelfall können Sie diese Warnungen einfach ignorieren und den Button *Fortsetzen* anklicken.)

Mailbox-Interna: KMail speichert E-Mails im Verzeichnis `~/Mail`. Die Default-Verzeichnisse für den Posteingang, -ausgang, für gesendete Nachrichten etc. (`inbox`, `outbox`, `sent-mail`) haben das mbox-Format. Beim Erzeugen neuer E-Mail-Verzeichnisse können Sie zwischen dem mbox- und dem maildir-Format wählen, wobei per Default das maildir-Format eingesetzt wird. Das Default-Format können Sie im Konfigurationsdialog *Ordner* einstellen.) Das Format vorhandener Verzeichnisse kann nicht geändert werden. Um ein Mail-Verzeichnis von einem Format in das andere umzuwandeln, erzeugen Sie ein neues Verzeichnis und kopieren alle Nachrichten vom alten Verzeichnis in das neue.

Zu allen mbox-Verzeichnissen legt KMail jeweils drei Indexdateien an, deren Name mit `.name.index` beginnt. Bei Verzeichnishierarchien erzeugt KMail für jedes Unterverzeichnis einen Ordner mit dem Namen `.name.directory`. Verzeichnisse können nicht per Drag&Drop in andere Verzeichnisse verschoben werden. Stattdessen öffnen Sie den Eigenschaftsdialog des Verzeichnisses und stellen im Listenfeld *Gehört zu* das übergeordnete Verzeichnis ein.

E-Mails verschieben und löschen: Jedes Mal, wenn Sie eine E-Mail per Drag&Drop in einen anderen Ordner verschieben, fragt KMail, ob die E-Mail verschoben oder kopiert werden soll. Diese Rückfrage vermeiden Sie, wenn Sie gleichzeitig die Taste (Shift) (verschieben) bzw. (Strg) (kopieren) drücken oder

wenn Sie im Konfigurationsdialog *Ordner* die Option *Beim Verschieben von Nachrichten fragen* deaktivieren.

Entf verschiebt die markierten E-Mails in das Verzeichnis *Mülleimer*. Wenn Sie die E-Mail endgültig löschen möchten, verwenden Sie stattdessen Shift+Entf. Wenn Sie möchten, kann der *Mülleimer* beim Beenden von KMail automatisch geleert werden (Konfigurationsdialog *Ordner*, Option *Mülleimer leeren*).

E-Mail-Formatierung (HTML-Mail): KMail bietet keine Möglichkeit, HTML-Mails zu verfassen. Auch eintreffende E-Mails werden möglichst in Textform angezeigt. Bei E-Mails, die auch in HTML-Form vorliegen, gelangen Sie über einen Link innerhalb der Nachricht zur HTML-Darstellung. Wenn Sie generell die HTML-Darstellung bevorzugen, können Sie dies im Konfigurationsdialog *Sicherheit* einstellen. In diesem Dialog können Sie weiter angeben, dass auch externe Bilder angezeigt werden (*externe Referenzen laden*). Beide Einstellungen sind aus Sicherheitsgründen per Default deaktiviert.

Thread-Ansicht: Bei E-Mails aus Mail-Listen kann durch die Thread-Ansicht die Struktur des Nachrichtenverkehrs nachvollzogen werden. Diese Ansicht aktivieren Sie mit *Ordner → Nachrichten gruppiert anzeigen*.

Webbrowser: Wenn Sie einen Link anklicken, zeigt KMail die Webseite automatisch mit Konqueror an. Sie können dieses Verhalten nur im KDE-Kontrollzentrum (`kcontrol`) ändern, indem Sie im Dialog *KDE → Dateizuordnungen* für das Dateimuster `html` das auszuführende Programm ändern. Diese Einstellung gilt für alle KDE-Programme.

Filter für den Posteingang: Wenn Sie viele E-Mails aus Mailing-Listen erhalten, kann es zweckmäßig sein, eintreffende E-Mails automatisch auf verschiedene Verzeichnisse zu verteilen. Solche Filter können Sie mit *Extras → Filter anlegen* einrichten. Sie sollten vorher eine betreffende E-Mail markieren – KMail hilft dann gleich beim Erstellen der Filterregel. Um den neuen Filter auszuprobieren, markieren Sie mit K alle Nachrichten des aktuellen Verzeichnisses und führen dann *Nachricht → Filter anwenden* aus.

Neben den hier beschriebenen Filtern können Sie auch POP3-Filter definieren, die angewendet werden, noch bevor die E-Mail vom POP-Server geladen wird (*Extras → POP-Filter einrichten*). Beachten Sie, dass KMail eintreffende E-Mails grundsätzlich nicht filtern kann, wenn diese von einem IMAP-Account stammen. (Die Hintergründe sind in den KMail-FAQs genau beschrieben.)

Spam-Filter: In KMail ist kein eigener Spam-Filter integriert. KMail sieht aber Mechanismen für eine einfache Integration von externen Spam-Filtern wie *Bogofilter* oder *SpamAssassin* vor. Links zu einer Reihe von Anleitungen finden Sie unter *http://kmail.kde.org/tools.html*.

5.4 Zugriff auf Linux-Verzeichnisse im Netz (NFS)

In einem Netzwerk möchte man oft auf die Dateien anderer Rechner zuzugreifen. (Oft ist genau das der Grund, überhaupt ein Netzwerk einzurichten.) Dieser Abschnitt beschreibt hierfür einen Unix- bzw. Linux-typischen Weg: das Einbinden von NFS-Netzwerkverzeichnissen in das lokale Dateisystem. Damit erreichen Sie, dass Sie auf externe Dateien ebenso zugreifen können wie auf lokale. Eine Alternative zum *Network File System* (NFS) ist das in der Microsoft-Welt übliche *Server Message Block*-Protokoll (SMB), das Sie unter Linux ebenfalls nutzen können.

5.4.1 NFS-Client

Damit Sie NFS verwenden können, müssen einige Voraussetzungen erfüllt sein: Der Rechner, dessen Dateien Sie lesen oder verändern möchten, muss über das Netz erreichbar sein (testen Sie das mit `ping`). Außerdem muss dort ein NFS-Server installiert sein, und dieser Server muss so konfiguriert sein, dass Sie die Erlaubnis zum Dateizugriff auf das gewünschte Verzeichnis haben.

Sind diese drei Voraussetzungen erfüllt, ist der Zugriff auf ein NFS-Verzeichnis denkbar einfach. Ein einfaches `mount`-Kommando genügt:

```
root# mount -t nfs schrottsocke:/data /externaldata
```

Damit wird das Verzeichnis `/data` des Rechners `schrottsocke` auf dem lokalen Rechner unter dem Verzeichnis `/externaldata` verfügbar. (Das Verzeichnis `/externaldata` muss natürlich existieren, bevor `mount` ausgeführt wird!) Von jetzt an können Sie auf alle Daten in `schrottsocke:/data` so zugreifen, als würden sie sich auf dem lokalen Rechner befinden. Ob Sie die Daten auch verändern dürfen, hängt von den Konfigurationseinstellungen auf dem NFS-Server ab. Statt des Rechnernamens können Sie auch die IP-Nummer angeben – also beispielsweise `10.10.10.1:/data`. Das ist eine Notlösung, wenn kein lokaler Nameserver existiert und der Name des NFS-Servers nicht in `/etc/hosts` eingetragen ist.

Hinweis

Wenn das `mount`-Kommando bei Red Hat nicht funktioniert, ist die wahrscheinlichste Ursache eine zu restriktive Firewall. Wenn Sie die Firewall während der Installation eingestellt haben, darf am Client maximal die Stufe *mittel* verwendet werden; am Server muss die lokale Netzwerkschnittstelle als *sicheres Gerät* gekennzeichnet sein.

Mit `umount` wird das NFS-Verzeichnis wieder aus dem lokalen Dateisystem entfernt. Wenn die Netzwerkverbindung gerade unterbrochen ist, sollten Sie `umount` mit der Option `-f` (force) ausführen. Sonst müssen Sie lange warten, bis `umount` ausgeführt wird!

```
root# umount /externaldata
```

mount-Optionen: Es gibt eine Menge NFS-spezifischer `mount`-Optionen:

- Die Option **hard** bewirkt, dass ein Programm, das auf eine NFS-Datei zugreift, hängen bleibt, wenn der NFS-Server nicht mehr zur Verfügung steht.
- Die Option **soft** bewirkt, dass der Kernel es nach einer Weile aufgibt, die NFS-Datei wiederzufinden. Das klingt zwar sicherer, verursacht aber in der Realität noch mehr Probleme als „hard".
- Dank der Option **intr** kann ein Programm auch dann durch `kill` bzw. Strg+C gestoppt werden, wenn eine offene NFS-Datei nicht mehr zur Verfügung steht. Die Option ist nur in Kombination mit „hard" gültig.
- Die Optionen **rsize=***n* und **wsize=***n* geben den Puffer für Lese- und Schreiboperationen in KByte an (default 4096).

Tipp

Die richtigen `mount`-Optionen entscheiden maßgeblich über die mit NFS erzielbare Geschwindigkeit! So steigt die reale Geschwindigkeit bei einem 100-MBit-Netzwerk durch `hard,intr,rsize=8192,wsize=8192` auf rund 9 MByte/s, was dem theoretischen Maximum schon recht nahe kommt (100 MBit/s = 12,5 MByte/s). Ohne diese Optionen betrug die Geschwindigkeit bei meinen Messungen rund 7 MByte/s. Die Benchmark-Tests waren zugegebenermaßen nicht sehr ausgeklügelt. Ich habe einfach einen 300 MByte großen Verzeichnisbaum per NFS kopiert. Wenn die NFS-Geschwindigkeit unter Ihren Erwartungen liegt, stellen Sie sicher, dass `/etc/exports` am NFS-Server die Option `async` verwendet.

Selbstverständlich können Sie NFS-Verzeichnisse auch in `/etc/fstab` eintragen. In der vierten Spalte können Sie die NFS-spezifische Option `bg` verwenden. Sie erreichen damit, dass `mount` im Hintergrund versucht, das Netzwerkverzeichnis einzubinden, wenn dieses nicht sofort zur Verfügung steht. Das ist vor allem beim Einbinden von Netzwerkverzeichnissen beim Rechnerstart praktisch.

```
# Ergänzung in /etc/fstab
jupiter:/data    /externaldata  nfs  user,exec,bg  0  0
```

Bei den meisten Distributionen werden alle in `/etc/fstab` genannten Verzeichnisse während des Init-V-Prozesses eingebunden. Eine Ausnahme stellt Suse dar: Hier ist für diesen Vorgang das Init-V-Script `nfs` zuständig, das explizit mittels `insserv nfs` aktiviert werden muss.

5.4.2 NFS-Server

Bei den meisten Distributionen ist es nicht nötig, einen neuen Linux-Kernel zu kompilieren, um NFS benutzen zu können – es müssen lediglich das NFS-Dateisystem und NFS-Unterstützung aktiviert werden. Um einen NFS-Server einzurichten, müssen lediglich mit den Skripten `/etc/rc.d/rpc` und `/etc/rc.d/nfsserver` der Portmapper und der NFS-Dämon (nfsd) gestartet werden.

Die einzige Konfigurationsdatei für den NFS-Dämon ist die Datei `/etc/exports` mit folgendem Format:

```
<Verzeichnis-Pfad>     <Rechnernamen (Optionen)>
```

Links steht das Verzeichnis, das der NFS-Server exportieren soll, beispielsweise `/home/public` oder `/data`. In der Mitte stehen die Rechner, die Zugriff auf das Verzeichnis haben sollen, und danach in Klammern die Optionen. Beachten Sie das Leerzeichen zwischen den Rechnernamen und den Optionen! Nach jeder Änderung der Datei müssen Sie den Portmapper neu starten und dann den NFS-Dämon mittels der Option `reload` die Konfigurationsdatei neu einlesen lassen.

Die zugriffsberechtigten Client-Rechner können Sie auf drei Arten definieren:

- Ein einzelner Rechnername oder eine einzelne IP-Nummer. Damit gestatten Sie nur diesem einen Rechner den Zugriff auf das Verzeichnis. Wenn Sie den Rechnernamen angeben, sollte in der Datei `/etc/hosts` seinem Namen eine IP-Adresse zugeordnet sein.
- Domain-Eintrag mit Jokerzeichen (* oder ?). Damit können Sie allen Rechnern einer Domain den Zugriff gestatten, z. B.: `*.mydomain.athome`
- IP-Netzwerknummern. Durch Eingabe eines Subnetzes mit Netzmaske. Wenn Sie z. B. `192.168.253.255/255.255.255.255` angeben, haben alle Rechner im Subnetz `192.168.253.0` Zugriff auf das Verzeichnis.

Die gebräuchlichsten Optionen sind:

- rw: Read-Write gibt den Clients Lese- und Schreibrechte im Verzeichnis.
- ro: Read-Only gibt den Clients nur Leserecht (Voreinstellung).
- noaccess: Verbietet Clients den Zugriff auf Unterverzeichnisse
- root-squash: Dateien mit User/Group `root` werden bei den Clients einem anonymen Eigentümer und einer anonymen Gruppe zugeordnet.
- no-root-squash: Das Gegenteil zu obiger Option
- async: siehe oben

Im folgenden Beispiel sollen folgende Zugriffe möglich sein: Auf die erste CD-ROM bekommen die Clients nur Lesezugriff. Der Rechner `schrottsocke.netzmafia.de` benötigt root-Zugriff auf `/install` `spielkiste.netzmafia.de` darf ebenfalls auf `/install` zugreifen, allerdings ohne daß Dateien des Benutzers `root` als solche erscheinen. Die Home-Verzeichnisse aller Benutzer auf dem Server exportiert der NFS-Server an alle Rechner im Subnetz, damit die Benutzer auf allen Clients das gleiche Home-Verzeichnis bekommen:

```
# /etc/exports
#
/cdrom     *.netzmafia.de(ro)
/install   schrottsocke.netzmafia.de(rw,no-root-squash) \
           spielkiste.netzmafia.de(ro,root-squash)
/home      192.168.253.255/255.255.255.255
```

Mit den folgenden Kommandos (Shell-Skripte) haben Sie die Möglichkeit der Fehlersuche

- `/etc/rc.d/rpc status`,
- `/etc/rc.d/nfsserver status` und
- `rpcinfo -p`

5.5 Zugriff auf Windows-Verzeichnisse im Netz (SMB)

Das in der Windows-Welt übliche *Server Message Block*-Protokoll (SMB) ist das Gegenstück zu NFS für Unix/Linux. Damit Sie SMB unter Linux nutzen können, müssen Sie Samba installieren. Samba ist ein ganzes Bündel von Programmen, dessen Hauptaufgabe eigentlich darin besteht, anderen (Windows-)Rechnern Zugriff auf Dateien und Drucker zu geben, die von Linux verwaltet werden.

Dieser Abschnitt widmet sich Samba aus Client-Sicht: Sie erfahren also, wie Sie mit Ihrem Linux-Rechner auf Windows- bzw. Samba-Ressourcen anderer Rechner zugreifen können. Grundsätzlich gibt es drei Möglichkeiten, Dateien aus Windows-Verzeichnissen unter Linux anzusprechen: durch SMB-kompatible Dateimanager, durch das Programm `smbclient` sowie durch das Netzwerkdateisystem `smbfs`. Bei vielen Distributionen ist `samba` in Client- und Server-Pakete zerlegt. Für den angepeilten Zweck reicht die Installation der Client-Pakete aus.

Die bequemste Art des SMB-Datenzugriffs bieten Dateimanager, die das SMB-Protokoll direkt unterstützen. Dies ist beispielsweise beim Konqueror (KDE) bzw. bei Nautilus (Gnome) der Fall. Um nach Windows- bzw. Samba-Servern im Netz zu suchen, geben Sie in der Adressleiste entweder „smb:/“ (Konqueror) oder „smb:///“ (Nautilus) ein. (Bei SUSE kann Konqueror erst nach der Installation des Pakets `kdebase3-samba` als Samba-Browser verwendet werden.)

Um auf das Verzeichnis `myshare` auf dem Windows-Rechner `venus` zuzugreifen, verwenden Sie bei Konqueror „smb://venus/myshare“ und bei Nautilus „smb:///venus/myshare“.

Wenn Sie nicht unter KDE arbeiten, können Windows-Ressourcen auch mit dem Programm `LinNeighborhood` erforscht werden. Damit das Programm wunschgemäß funktioniert, müssen je nach Netzwerkkonfiguration vorher zwei Parameter eingestellt werden:

- *Ändern* → *Voreinstellungen* → *Arbeitsgruppe*: Hier geben Sie den Namen der Windows-Arbeitsgruppe an.
- *Ändern* → *Voreinstellungen* → *Primärer Master-Browser*: In seltenen Fällen müssen Sie hier den Rechnernamen eines Windows- oder Samba-Servers angeben.

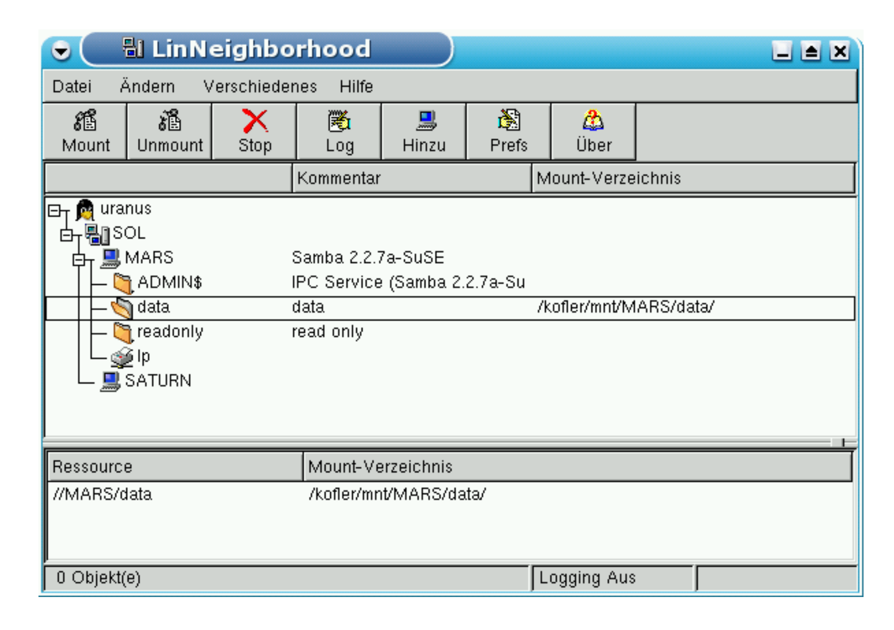


Abbildung 5.4: Windows-Verzeichnisse mit LinNeighborhood erforschen

Nun liefert ein Doppelklick auf den lokalen Rechnernamen (`uranus` eine Liste der anderen Rechner im Netz. Ein weiterer Doppelklick auf einzelne Netzwerkverzeichnisse bindet diese in das lokale Dateisystem ein. Gegebenenfalls müssen Sie dazu einen SMB-Benutzernamen und das dazugehörende Passwort angeben. Damit `mount` ausgeführt werden kann, benötigt das Programm außerdem das `root`-Passwort.

smbclient

Das Textkommando `smbclient` ist ähnlich zu bedienen wie ein FTP-Client. Mit `smbclient -L` *name* werden alle freigegebenen Ressourcen des Rechners *name* angezeigt.

```
user$  smbclient -L venus
added interface ip=192.168.0.2 bcast=192.168.0.255 nmask=255.255.255.0
Password:  ******
Domain=[SOL] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]

        Sharename      Type      Comment
        ---------      ----      -------
        HP III         Printer
        myshare        Disk

        Server               Comment
        ---------            -------
        VENUS

        Workgroup            Master
        ---------            -------
        SOL                  VENUS
```

Wenn `smbclient` eine Login-Fehlermeldung liefert (*access denied*), liegt das im Regelfall daran, dass die Benutzer- oder Workgroup-Namen Ihres Linux-Rechners nicht mit denen des Windows-Rechners oder Samba-Servers übereinstimmen. Die einfachste Lösung besteht darin, diese Informationen einfach als zusätzliche Parameter an `smbclient` zu übergeben. (Noch mehr Konfigurationsmöglichkeiten eröffnet die Datei `/etc/smb.conf`. Diese Datei dient zwar primär zur Konfiguration eines Samba-Servers, einzelne Einstellungen werden aber auch von `smbclient` ausgewertet.)

```
user$  smbclient -U benutzername -W workgroupname -L venus
```

Mit `smbclient //venus/myshare` stellen Sie eine Verbindung zum Rechner `venus` für das freigegebene Verzeichnis `myshare` her. Dabei werden Sie nach einem Passwort gefragt. Anschließend können Sie wie bei FTP Verzeichnisse mit `ls` ansehen, mit `cd` wechseln, mit `get` Dateien auf den lokalen Rechner übertragen (*download*) und mit `put` Dateien auf dem externen Rechner speichern (*upload*). Einen Überblick über die wichtigsten Kommandos bekommen Sie mit `help` oder auf derzugehörigen Man-Page.

```
user$  smbclient //venus/myshare -U name -W wgname
added interface ip=192.168.0.2 bcast=192.168.0.255 nmask=255.255.255.0
Password:  xxxxxx
Domain=[SOL] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: >  ls
  .                  D        0  Thu Sep  7 17:38:02 2000
```

```
  ..              D        0  Thu Sep  7 17:38:02 2000
  data            D        0  Wed Apr  5 18:17:11 2000
  file.xy        AR      226  Sat Dec 14 00:00:00 2000
```

smbfs-Dateisystem

Die auf den ersten Blick eleganteste Variante besteht darin, ein externes Verzeichnis wie bei NFS in das lokale Dateisystem einzubinden. Leider genießen Windows-Share-Verzeichnisse keinen besonders guten Ruf, was die Stabilität betrifft.

Um ein externes Verzeichnis einzubinden, geben Sie eines der beiden folgenden Kommandos an (je nachdem, ob die Windows-Freigabe auf der Basis von Benutzernamen erfolgt oder nicht):

```
root#  mount -t smbfs //venus/myshare /extdata
root#  mount -t smbfs -o username=name //venus/myshare /extdata
```

Damit wird das auf dem Windows-Rechner `venus` unter dem Namen `myshare` freigegebene Verzeichnis in das Linux-Dateisystem eingebunden. Die Daten stehen jetzt unter dem Linux-Verzeichnis `/extdata` zur Verfügung. (Dieses Verzeichnis muss vor dem Ausführen von `mount` natürlich schon existieren.)

Bei der Ausführung des Kommandos werden Sie nach dem Passwort gefragt. Sie können das Passwort aber auch direkt angeben. Das Kommando ist hier nur aus Platzgründen über zwei Zeilen verteilt.

```
root#  mount -t smbfs -o username=name,password=pw \
           //venus/myshare /extdata
```

Hinweis

Die Passwort-Abfrage erscheint selbst dann, wenn gar kein Passwort erforderlich ist. Sie können die Abfrage vermeiden, indem Sie die Option `passwort=` ohne Angabe eines Passworts verwenden – dann gilt eine leere Zeichenkette als Passwort. Vergessen Sie das Zeichen `=` nicht!

Intern führt `mount` das Kommando `smbmount` aus. Auf dessen Manual-Seite finden Sie daher eine Beschreibung der weiteren Optionen, die bei `mount` angegeben werden können. Selbstverständlich ist auch ein entsprechender Eintrag in `/etc/fstab` möglich.

```
# Ergänzung in /etc/fstab
//venus/myshare    /extdata  smbfs  user,noauto,exec  0  0
```

5.6 FTP-Client

FTP steht für *File Transfer Protocol* und bezeichnet eines der ersten Verfahren zur Übertragung von Dateien über ein Netzwerk. Seine große Popularität verdankt FTP der Spielart *Anonymous FTP*, bei der als Username „ftp“ oder „anonymous“ ausreicht. Dieser Zugang ist auch nicht durch ein Passwort versperrt. Leider werden beim FTP-Login-Prozess der Benutzername und das Passwort unverschlüsselt übertragen. Eine sichere Alternative ist SFTP (*Secure FTP*) auf der Basis von SSH (siehe unten). Damit FTP funktioniert, muss auf der Gegenstelle ein FTP-Server laufen.

Hinweis

Viele FTP-Clients haben Probleme, wenn sich zwischen ihrem Rechner und dem FTP-Server eine Firewall befindet oder wenn sie in einem lokalen Netzwerk arbeiten, das mittels Masquerading mit dem Internet verbunden ist. In solchen Fällen hilft es fast immer, den Client in einen so genannten passiven Modus zu versetzen. Leider gibt es dafür kein einheitliches Kommando – werfen Sie also einen Blick in die Dokumentation! (Die meisten Clients erkennen derartige Situationen selbstständig und aktivieren den passiven Modus automatisch.)

5.6.1 FTP-Clients

Die Anzahl der Programme, die Sie zur Nutzung von FTP verwenden können, ist riesig:

- `ftp`: Der Urahn aller FTP-Clients, der auch bei Windows verfügbar ist. Die Benutzeroberfläche ist zwar spartanisch, aber dafür kann das Programm auch im Textmodus verwendet werden.
- `ncftp`: Diese Alternative zu `ftp` hat zwar ebenfalls eine textbasierte Benutzeroberfläche, ist aber komfortabler zu bedienen.
- `sftp`: Dieses Programm ist ähnlich minimalistisch wie `ftp`, aber dafür deutlich sicherer. Allerdings muss an der Gegenstelle ein SSH-Server laufen (kein FTP-Server).
- `mirror`, `wget`, `rsync`: Hierbei handelt es sich um Script-Programme zur rekursiven Übertragung ganzer (FTP-)Verzeichnisbäume.
- `gftp` und `kbear`: Diese FTP-Clients mit Gnome- bzw. KDE-Oberfläche werden so ähnlich bedient wie der beliebte Windows-FTP-Client `wsftp_le`: Zwei nebeneinander liegende Fensterbereiche zeigen je ein Verzeichnis des lokalen Rechners und des FTP-Servers.

- Webbrowser, Dateimanager: Alle unter Linux verfügbaren Webbrowser und Dateimanager können auch zum FTP-Download verwendet werden. Viele Anwender werden dabei gar keinen Unterschied zu einem gewöhnlichen HTTP-Download bemerken.

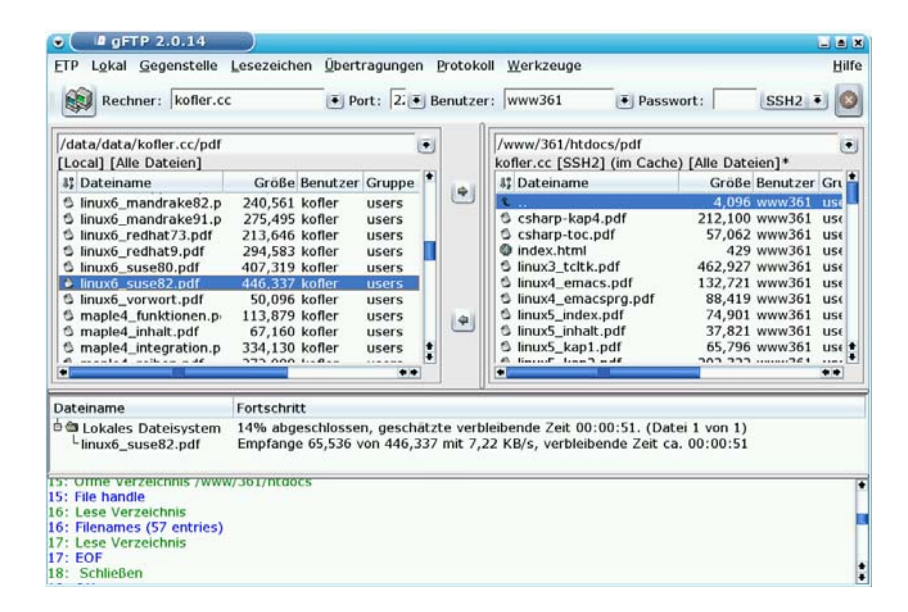


Abbildung 5.5: Datei-Upload mit gftp

Die meisten dieser Programme sind so intuitiv zu bedienen, dass hier auf eine Beschreibung verzichtet wird.

> **Tipp**
> Wenn Sie einen FTP-Server nicht als Benutzer `anonymous` verwenden möchten, kennen die meisten FTP-Clients die Syntax „ftp://benutzername@servername“. Sie werden automatisch zur Angabe des Passworts aufgefordert.

5.6.2 FTP-Kommando

Da Dateien per Default aus dem bzw. in das aktuelle Verzeichnis übertragen werden, sollten Sie vor dem Start von `ftp` mit `cd` in das gewünschte Arbeitsverzeichnis wechseln. Die FTP-Sitzung wird dann mit dem Kommando `ftp user@ftpservername` oder einfach `ftp ftpservername` eingeleitet. Falls Sie Anonymous FTP nutzen möchten, geben Sie als Benutzername `anonymous` oder `ftp` ein.

Mit den Kommandos `cd`, `pwd` und `ls`, die dieselbe Bedeutung wie unter Linux haben, können Sie sich durch die Verzeichnisse des FTP-Archivs bewegen. Um eine Datei vom FTP-Archiv in das aktuelle Verzeichnis Ihres Rechners zu übertragen, führen Sie `get datei` aus. Der Dateiname bleibt dabei unverändert. Umgekehrt können Sie mit `put` eine Datei aus Ihrem aktuellen Verzeichnis in ein Verzeichnis des FTP-Archivs übertragen. (nur, wenn Sie eine Schreiberlaubnis für das Verzeichnis haben).

Eine FTP-Sitzung wird mit dem Kommando `quit` oder `bye` beendet. Die folgende Tabelle fasst die wichtigsten Kommandos zusammen, die während einer FTP-Sitzung ausgeführt werden können:

ftp-Kommandos	
`?`	zeigt eine Liste aller FTP-Kommandos an
`!`	ermöglicht die Ausführung von Shell-Kommandos
`ascii`	wechselt in den Textmodus
`binary`	wechselt in den Binärmodus
`bye`	beendet FTP
`cd verz`	wechselt in das angegebene FTP-Verzeichnis
`close`	beendet die Verbindung zum FTP-Server
`get datei`	überträgt die Datei vom FTP-Archiv in das aktuelle Verzeichnis
`help kommando`	zeigt eine kurze Info zum angegebenen Kommando an
`lcd verz`	wechselt das aktuelle Verzeichnis am lokalen Rechner
`ls`	zeigt die Liste der Dateien am FTP-Server an
`lls`	zeigt die Liste der Dateien am lokalen Rechner an
`mget *.muster`	überträgt alle passenden Dateien vom FTP-Archiv in das aktuelle Verzeichnis (siehe auch `prompt`)
`open`	stellt die Verbindung zum fremden Rechner her (wenn es beim ersten Versuch nicht geklappt hat)
`prompt`	aktiviert/deaktiviert die automatische Rückfrage vor der Übertragung einer jeden Datei durch `mget`
`put datei`	überträgt die Datei vom aktuellen Verzeichnis in das FTP-Archiv
`quit`	beendet FTP
`reget datei`	setzt die Übertragung einer bereits teilweise übertragenen Datei fort
`user`	ermöglicht einen neuen Login (wenn es beim ersten Versuch nicht funktioniert hat)

Bevor Sie eine Datei übertragen, müssen Sie mit `binary` in den Binärmodus umschalten. Im Textmodus interpretiert FTP die Dateien als Texte und versucht, diese in das Format des jeweiligen Rechners zu konvertieren (hauptsächlich Ersetzen von Carriage Return und Linefeed durch Linefeed und umgekehrt). Binärdateien werden durch solch eine Konvertierung unbrauchbar.

5.7 SSH

Die Programme `telnet`, `rlogin` und `ssh` ermöglichen es, auf einem anderen Rechner zu arbeiten, als stünde er vor Ihnen. Das funktioniert sowohl für kommando-orientierte Programme als auch für X-Programme.

Dieser Abschnitt beschränkt sich auf die Beschreibung von `ssh` (*secure shell*). Die älteren Programme `telnet` und `rlogin` sollten aus Sicherheitsgründen nicht mehr eingesetzt werden. Sie übertragen alle Informationen, also auch die Login-Informationen, unverschlüsselt. Die Grundvoraussetzung für die Anwendung von `ssh` besteht darin, dass am zweiten Rechner ein SSH-Server läuft, also das Programm `sshd`. Bei vielen Linux-Distributionen ist dies mittlerweile per Default der Fall.

Gewöhnliche Shell-Session: Wenn Sie auf dem Rechner `spielkiste` arbeiten und nun eine Shell-Session auf dem Rechner `schrottsocke` starten möchten, führen Sie zum Verbindungsaufbau das folgende Kommando aus:

```
user@spielkiste$  ssh schrottsocke
user@schrottsocke's password:  xxx
```

Wenn Sie mit `ssh` zum ersten Mal eine Verbindung zu einem anderen Rechner herstellen, erscheint eine Warnung nach dem folgenden Muster:

```
The authenticity of host 'schrottsocke (10.10.10.3)' can't be established.
RSA1 key fingerprint is 1e:0e:15:ad:6f:64:88:60:ec:21:f1:4b:b7:68:f4:32.
Are you sure you want to continue connecting (yes/no)?  yes
Warning: Permanently added 'schrottsocke,10.10.10.3' (RSA1) to the list
  of known hosts.
```

Das bedeutet, dass `ssh` sich nicht sicher ist, ob es dem Rechner `schrottsocke` mit der IP-Adresse 10.10.10.3 vertrauen darf; oder es kennt den Rechner noch nicht. Wenn Sie die Rückfrage mit *yes* beantworten, speichert `ssh` den Namen, die Adresse und den RSA-Fingerprint (einen Code zur eindeutigen Identifizierung des Partnerrechners) in `~/.ssh/known_hosts`.

Falls Sie auf `schrottsocke` unter einem anderen Login-Namen als auf `spielkiste` arbeiten möchten (z. B. als `root`), geben Sie den Namen mit der Option `-l` oder per Klammeraffe an (`ssh -l root schrottsocke` oder `ssh root@schrottsocke`).

Kommandos ausführen: Statt `ssh` interaktiv zu nutzen, können Sie am entfernten Rechner auch einfach nur ein Kommando ausführen. Das Kommando und seine Parameter werden einfach als weitere Parameter an `ssh` übergeben. `ssh` endet nach diesem Kommando.

```
user@spielkiste$  ssh schrottsocke kommando optionen
user@schrottsocke's password:  xxx
```

Aus dieser scheinbar trivialen Funktion ergeben sich weit reichende Möglichkeiten. Sie können nun beispielsweise am entfernten Rechner `tar` starten, das damit erstellte Archiv an die Standardausgabe weiterleiten (Bindestrich - nach der Option `-f`) und diese Ausgabe als Eingabe für ein zweites `tar`-Kommando verwenden, das lokal läuft. Damit können Sie einen ganzen Verzeichnisbaum sicher via SSH kopieren.

Das folgende Kommando zeigt, wie ich den gesamten `htdocs`-Verzeichnisbaum des Webservers `kofler.cc` in das lokale Verzeichnis `~/bak` kopiere. Das Kommando wurde nur aus Platzgründen mit \ auf zwei Zeilen verteilt.

```
user$  ssh -l username kofler.cc tar -cf - htdocs — \
         tar -xC ~/bak/ -f -
username@kofler.cc's password:  ******
```

SSH und X: Grundsätzlich kann in einer SSH-Session auch ein X-Programm ausgeführt werden. Das Programm läuft dann auf dem entfernten Rechner, wird aber auf dem lokalen Rechner angezeigt und empfängt dort auch alle Tastatur- und Mauseingaben. Damit `ssh` X-Programme ausführen kann, muss es mit der Option `-X` gestartet werden. (Wenn `/etc/ssh_config` die Zeile „ForwardX11 yes" enthält, kann auf die Option `-X` verzichtet werden.) `ssh` kümmert sich selbstständig um die korrekte Einstellung der `DISPLAY`-Variablen.

Die folgenden Kommandos bewirken, dass am Rechner `schrottsocke` der Editor XEmacs gestartet wird. Das Editorfenster wird aber auf dem Desktop des Rechners `spielkiste` sichtbar und kann dort bedient werden! Das funktioniert selbst dann, wenn auf dem Rechner `schrottsocke` gar kein X-Server läuft (Runlevel 3). Alle X-Bibliotheken müssen aber installiert sein!

```
user@spielkiste$  ssh -X schrottsocke
user@schrottsocke's password:  xxx
user@schrottsocke$  xemacs &
```

Dateien sicher kopieren (scp): Um eine Datei via SSH über das Netzwerk zu kopieren, gibt es das Kommando `scp`. Die Syntax sieht so aus:

```
user$  scp [[user1@]host1:]filename1 [[user2@]host2:][filename2]
user2@host2's password:  ******
```

Damit wird die Datei `filename1` vom Rechner `host1` zum Rechner `host2` übertragen und dort in der Datei `filename2` gespeichert. Einige Anmerkungen zu den vielen optionalen Bestandteilen der Kopieranweisung:

- `host1` und `host2` müssen nicht angegeben werden, wenn der lokale Rechner (also `localhost`) gemeint ist.
- `user1` muss nicht angegeben werden, wenn der aktive Benutzer gemeint ist.
- `user2` muss nicht angegeben werden, wenn am Rechner `host2` der aktuelle Benutzername von `host1` bzw. `user1` verwendet werden soll.

- `filename1` darf auch ein Verzeichnis sein. Sie müssen dann die Option `-r` angeben, damit das gesamte Verzeichnis mit allen Unterverzeichnissen übertragen wird.
- `filename2` muss nicht angegeben werden, wenn der Dateiname unverändert bleiben soll. Die Datei wird dann in das Home-Verzeichnis von `user2` kopiert.

  Statt `filename2` kann auch das Zielverzeichnis angegeben werden, wobei wie üblich `~` für das Home-Verzeichnis von `user2` verwendet wird.

Eine SSH-Anwendungsmöglichkeit für fortgeschrittene Linux-Anwender ist der **SSH-Tunnel**. Er macht die Übertragung aller IP-Pakete, die an einen bestimmten Port gerichtet sind, über SSH möglich. SSH-Tunnel bieten damit einen sicheren Weg, um IP-Pakete zwischen zwei Rechnern zu übertragen – und das selbst dann, wenn sich zwischen den beiden Rechnern eine Firewall befindet, die den Port eigentlich blockiert.

Erfolgt der Tunnelbau vom Client-Rechner aus, kommt die Option `-L localport:remotehost:remoteport` zum Einsatz. Beispielsweise bewirkt das folgende Kommando, dass der Port 3306 des Rechners `schrottsocke` über den Port 3307 des lokalen Rechners zugänglich ist. Durch das Kommando wird gleichzeitig eine SSH-Session gestartet, was Sie durch `-N` aber verhindern können (wenn Sie nur den Tunnel, aber keine Shell benötigen). Falls der Login bei `schrottsocke` unter einem anderen Namen erfolgen soll, müssen Sie den Login-Namen wie üblich durch `-l name` angeben.

```
user@spielkiste$  ssh -L 3307:schrottsocke:3306 schrottsocke
user@schrottsocke's password:  ******
```

Der Tunnel bleibt so lange offen, bis die SSH-Session mit Strg+D beendet wird. Falls Sie `ssh` mit der Option `-N` gestartet haben, muss das Programm mit Strg+C gestoppt werden.

3306 ist der übliche Port von MySQL. Sie können nun am Rechner `spielkiste` über dessen Port 3307 auf den MySQL-Server zugreifen, der auf `schrottsocke` läuft. Beim `mysql`-Kommando muss der Port 3307 und der Hostname 127.0.0.1 angegeben werden, damit der SSH-Tunnel tatsächlich benutzt wird. (per Default stellt `mysql` lokale Verbindungen über eine Socket-Datei her.)

```
user@spielkiste$  mysql -u mysqllogin -P 3307 -h 127.0.0.1 -p
Enter password:  ¡mysqlpassword¿
```

Damit der MySQL-Login funktioniert, müssen zwei Voraussetzungen erfüllt sein. Erstens muss der MySQL-Server am Rechner `schrottsocke` grundsätzlich IP-Verbindungen akzeptieren. Zweitens muss der MySQL-Server die Kombination aus Login-Name und Hostnamen akzeptieren. Als Hostname wird der Name des Rechners verwendet, zu dem `ssh` den Tunnel errichtet hat – hier also `schrottsocke` (bzw. `schrottsocke.mydomain`.

Es gibt noch weit mehr und oft viel komplexere Anwendungsmöglichkeiten für SSH-Tunnel. Beispielsweise können Sie die Tunnel dazu verwenden, um ein *Virtual Private Network* zu bilden.

5.7.1 SFTP (Secure FTP)

Das Kommando `sftp` ist Teil des `openssh`-Kommandos. `sftp` verwendet intern ein ganz anderes Protokoll als `ftp` und kann wie `ssh` nur eingesetzt werden, wenn auf der Gegenstelle ein SSH-Server läuft. Anonymous FTP ist mit `sftp` nicht möglich. Davon abgesehen, erfolgt die Bedienung des Programms aber genau wie die von `ftp`.

Vielen ist `sftp` zu spartanisch. Leider ist die Auswahl komfortablerer SFTP-Clients etwas geringer als bei FTP. Außerdem ist manchmal etwas Überredungskunst erforderlich, bis der Verbindungsaufbau klappt. `gftp` bietet vielseitige SFTP-Konfigurationsmöglichkeiten (*FTP* → *Optionen* → *SSH*). Wenn es Probleme gibt, achten Sie darauf, dass Sie den richtigen Port verwenden (22 für SSH, nicht 21 wie bei FTP). Häufig müssen Sie außerdem *Verwende SSH2 SFTP Funktionen* im Optionsdialog aktivieren. Auch kbear kommt mit SFTP zurecht.

Beim KDE-Universal-Browser initiieren Sie eine SFTP-Verbindung, indem Sie die Adresse `sftp://user@servername` eingeben. Nach der Passwortabfrage verhält sich Konqueror wie bei einem lokalen Verzeichnis. `konqueror` unterstützt aber auch direkt das SSH-Protokoll, das selbst dann funktioniert, wenn `sftp` nicht zur Verfügung steht. Dazu geben Sie die Adresse in der Form `fish://user@servername` an.

5.8 Verzeichnisse kopieren und synchronisieren (rsync)

Das Kommando `rsync` hilft dabei, ganze Verzeichnisbäume zu kopieren bzw. zu synchronisieren. Im Prinzip können Sie das natürlich auch mit `cp` machen, `rsync` hat aber einige wesentliche Vorteile:

- `rsync` ist besonders gut zur Synchronisierung von Verzeichnissen über (eventuell langsame) Netzwerke geeignet. Es werden nur die Änderungen an den Daten übertragen.
- Zur sicheren Datenübertragung kann `rsync` in Kombination mit SSH (*secure shell*) eingesetzt werden.
- `rsync` bietet zahlreiche Steuerungsoptionen, die weit über die Möglichkeiten von `cp` hinausgehen. Beispielsweise können Sie im Zielverzeichnis alle Dateien löschen, die es im Quellverzeichnis nicht mehr gibt.

Lokale Anwendung: Bei der lokalen Anwendung funktioniert `rsync` ähnlich wie `cp`. Der Hauptunterschied besteht darin, dass es unzählige Optionen gibt, mit denen das Kopierverhalten gesteuert werden kann.

Das folgende Kommando kopiert alle `*.tex`-Dateien von einem Verzeichnis in ein zweites. Anders als bei `cp` werden bereits vorhandene Dateien, die seit dem letzten Kopieren unverändert geblieben sind, nicht neuerlich kopiert.

```
user$  rsync verz1/*.tex verz2/
```

Um ein ganzes Verzeichnis mit allen Unterverzeichnissen zu synchronisieren, verwenden Sie die Option `-a`, die als Kurzschreibweise für eine ganze Reihe anderer Optionen gilt (`-rlptgoD`). Die Option bewirkt eine rekursive Verarbeitung aller Unterverzeichnisse und stellt sicher, dass möglichst alle Datei-Informationen (Besitzer, Gruppenzugehörigkeit, Zeitpunkt der letzten Änderung etc.) erhalten bleiben. Falls `verz2` noch nicht existiert, wird das Verzeichnis erzeugt.

```
user$  rsync -a verz1/ verz2/
```

per Default kopiert bzw. aktualisiert `rsync` alle neuen bzw. geänderten Dateien, löscht aber nichts. Wenn Sie möchten, dass aus `verz1` gelöschte Dateien oder Verzeichnisse auch in `verz2` gelöscht werden, geben Sie zusätzlich die Option `--delete` an.

Wenn Sie ein ausführliches Feedback erhalten möchten, welche Dateien verändert werden und wie viele Daten dazu in welcher Geschwindigkeit übertragen werden, geben Sie die zusätzliche Option `-v` an.

Anwendung im Netzwerk: Damit `rsync` auch eine Synchronisation zwischen vernetzten Rechnern durchführen kann, muss das Programm auch auf dem zweiten Rechner installiert sein. Es bestehen grundsätzlich zwei Möglichkeiten, wie die beiden `rsync`-Programme (also das lokale Kommando und das `rsync`-Programm des Partnerrechners) miteinander kommunizieren:

- Die `rsync`-Instanzen verwenden zur Kommunikation eine Shell, üblicherweise SSH (siehe Seite 247). Das hat den Vorteil, dass die Übertragung der Daten verschlüsselt erfolgt. Damit das funktioniert, muss auf beiden Rechnern SSH installiert sein; außerdem muss auf dem Partnerrechner ein SSH-Server so konfiguriert sein, dass SSH-Logins möglich sind.
- `rsync` läuft auf dem Partnerrechner als Server (Dämon). Der `rsync`-Server wird durch die Datei `/etc/rsyncd.conf` konfiguriert (siehe die entsprechende `man`-Seite) und durch ein Init-V-Script gestartet (in der Regel `/etc/init.d/rsyncd`). Dieses Client-Server-Szenario wird hier allerdings nicht weiter beschrieben.

Alle weiteren Beispiele setzen voraus, dass `rsync` via SSH kommuniziert. Dazu müssen Sie beim Aufruf des `rsync`-Kommandos die Option `-e ssh` verwenden.

Bevor Sie mit `rsync`-Experimenten beginnen, sollten Sie sicherstellen, dass ein gewöhnlicher SSH-Login zum Partnerrechner möglich ist.

Bei langsamen Netzwerkverbindungen kann die zusätzliche Option `-z` eingesetzt werden. Sie bewirkt die Komprimierung des `rsync`-Datenaustauschs. Das führt allerdings zu einer stärkeren CPU-Belastung auf beiden Rechnern, die leider nicht immer mit der erhofften schnelleren Synchronisierung verbunden ist.

Die Angabe der Quell- und Zielverzeichnisse erfolgt nun in der Schreibweise `hostname:verzeichnis` bzw. `username@hostname:verzeichnis`, falls nicht der aktuelle Benutzername verwendet werden soll.

Durch das folgende Kommando wird das Verzeichnis `verz1` des lokalen Benutzers `username` auf dem Rechner `spielkiste` mit dem Verzeichnis `verz2` auf dem Rechner `schrottsocke` synchronisiert. Für die Passworteingabe ist `ssh` verantwortlich. (Es muss also das Login-Passwort des Benutzers `username` am Rechner `schrottsocke` eingegeben werden.)

```
username@spielkiste$  rsync -e ssh -az verz1/ schrottsocke:verz2/
username@schrottsocke's password:  ******
```

`rsync` kann Dateien auch von einem entfernten Rechner auf den lokalen übertragen. Das folgende Kommando synchronisiert also in die umgekehrte Richtung:

```
username@spielkiste$  rsync -e ssh -az schrottsocke:verz2/ verz3/
username@schrottsocke's password:  ******
```

Kapitel 6

Drucken mit CUPS

Einführung

Dieses Kapitel beschreibt, wie Sie Ihren Drucker unter Linux konfigurieren und anschließend Dokumente ausdrucken. Generell ist die Druckerunterstützung unter Linux ausgezeichnet. Einzig bei einigen modernen Fotodruckern gibt es Probleme. Die meisten Distributionen liefern ausgereifte Konfigurationsprogramme mit, die das Einrichten eines Druckers zum Kinderspiel machen.

Das Kapitel geht auch etwas auf die Interna des Druckprozesses ein. Dabei wird insbesondere das Drucksystem CUPS berücksichtigt. In diesem Zusammenhang lernen Sie auch, wie Sie Netzwerkdrucker nutzen bzw. wie Sie Ihren Drucker anderen Netzwerkteilnehmern zur Verfügung stellen.

Die kommerziellen Druckertreiber der Firma TurboPrint werden hier nicht behandelt. Diese Treiber unterstützen manche Drucker, zu denen es momentan keine freien Treiber für Linux gibt. (Das betrifft insbesondere viele Farbtintenstrahldrucker von Canon.) Außerdem können Sie mit TurboPrint bei vielen Fotodruckern bessere Ergebnisse erzielen als mit den Standard-Treibern von CUPS. Sie finden den relativ preisgünstigen Treiber sowie eine kostenlose, aber eingeschränkte Free Edition auf der Website *http://www.turboprint.de/*

6.1 Konfiguration und Anwendung lokaler Drucker

Wenn Sie schon einen Drucker besitzen, müssen Sie sehen, wie Sie damit zurechtkommen. Wenn das aber noch nicht der Fall ist, lohnt sich vorher ein Blick auf die Webseiten *http://www.linuxprinting.org* und auch *http://www.linuxprinting.org/howto/*, wo Sie massenhaft Empfehlungen finden.

Es werden zwar schon recht viele, aber eben nicht alle Drucker von Linux unterstützt und zum Teil sogar automatisch erkannt.

> **Hinweis**
>
> Es gibt einige Drucker, die speziell für den Einsatz unter Windows entwickelt worden sind. Die Grundidee ist, dass ein Windows-Programm die gesamte zu druckende Seite zuerst auf dem Rechner vorbereitet und dann binär an den Drucker überträgt („Windows“- oder „GDI“-Drucker).
> Das Problem: Die gesamte Intelligenz sitzt im PC und Format zur Datenübertragung der Seite vom Rechner zum Drucker ist in den meisten Fällen nicht offen gelegt. Daher werden die meisten GDI-Drucker von Linux **nicht** unterstützt. Ausnahmen sind im oben erwähnten Printing-HOWTO dokumentiert.

Folgende Konfigurationstipps gelten für alle Distributionen:

- Bei manchen Druckermodellen stehen mehrere Treiber zur Auswahl (Siehe auch Seite 283). In solchen Fällen richtet man am besten den Drucker mehrfach unter verschiedenen Namen ein. Damit können Sie die verschiedenen Treiber bequem ausprobieren. Oft läßt sich auch nicht definitiv sagen, Treiber x ist besser als Treiber y. Die Qualität des Ausdrucks hängt stark davon ab, was Sie drucken möchten – Text, technische Zeichnungen, Fotos etc. Nach einigen Experimenten werden Sie wissen, welchen Druckernamen Sie am besten für welche Daten verwenden. Die Druckqualität wird auch ganz erheblich durch die Einstellung der Treiberparameter – z. B. der DPI-Auflösung – beeinflusst. Für den Fotodruck eignen sich die Gimp-Print-Treiber besonders gut. (Diese Treiber werden oft auch als stp-Treiber bezeichnet.)

  Die Treiber unterscheiden sich nicht nur durch die Qualität der Druckergebnisse, sondern auch durch die Druckgeschwindigkeit. Für den Ausdruck von Programmlistings ist nicht unbedingt ein Druckertreiber zum Fotodruck erforderlich!

- Falls Sie mehrere Drucker besitzen, sollten Sie während der Konfiguration eines Druckers alle anderen ausschalten. In vielen Fällen gelingt es dem Konfigurationsprogramm dann, den Drucker selbstständig zu erkennen.

- Bei der Konfiguration von Druckern stoßen Sie unweigerlich auf den Begriff *Warteschlange* (englisch: *queue*). Generell werden Drucker unter Linux nie direkt angesprochen, sondern über den Umweg einer Warteschlange (sonst gäbe es ein ziemliches Chaos, wenn mehrere User gleichzeitig drucken). Beim Starten eines Druckauftrags müssen Sie daher immer den Namen der Warteschlange angeben (in der Regel `lp`). Haben Sie nur einen einzigen Drucker konfiguriert, wird dieser mit dem Namen `lp` angesprochen. Die Warteschlange hat primär die Aufgabe, Druckjobs zwischenzuspeichern.

- Fast alle PostScript-Laserdrucker können auch in einem Kompatibilitätsmodus betrieben werden, so dass sie sich wie ein HP-Laserjet-Drucker der Serie

3, 4 oder 5 verhalten. Sie können also Ihren PostScript-Drucker zumeist auch als HP-Laserjet-kompatiblen Drucker konfigurieren. Das bewirkt, dass die Druckdaten von Ihrem Computer in das HP-Laserjet-Format umgewandelt und dann an den Drucker gesandt werden. Das wirkt zwar umständlich, ist aber in vielen Fällen mit einer (oft großen) Geschwindigkeitssteigerung verbunden. Der Grund: Für den Drucker ist die Verarbeitung von PostScript-Daten viel aufwändiger als die von Laserjet-Daten. Die Verarbeitungsgeschwindigkeit am Drucker ist aber viel kleiner als auf einem handelsüblichen PC. Natürlich hat diese Konfiguration auch einen Nachteil: Die Druckqualität ist zum Teil etwas geringer. Wie groß der Unterschied ist, hängt vor allem von den eingesetzten Schriften ab.

6.1.1 RedHat- und Fedora-Drucker-Konfiguration

Konfiguration: Zur Konfiguration des Druckers rufen Sie als `root` das Programm `system-config-printer` auf. Der Button *Neu* startet den Assistenten zur Drucker-Konfiguration. Im ersten Dialog geben Sie an, unter welchem Namen Sie den Drucker (genau genommen dessen Warteschlange) ansprechen möchten. Im Dialog *Warteschlangentyp* können Sie angeben, ob Sie auf einen lokalen Drucker oder einen Netzwerkdrucker zugreifen möchten. Bei lokalen Druckern werden alle in Frage kommenden Schnittstellen automatisch angezeigt, sofern der Drucker eingeschaltet ist.

Das Konfigurationswerkzeug sieht keine automatische Erkennung des Druckers vor. Deswegen müssen Sie im Dialog *Druckermodell* zuerst den Herstellernamen und dann das Modell manuell auswählen. Falls Sie Ihren Drucker nicht finden, müssen Sie ein kompatibles Gerät suchen oder eines der wenigen generischen Modelle auswählen (z. B. für PostScript-Drucker).

Damit ist die Basiskonfiguration beendet. Die neue Warteschlange wird gespeichert und Sie haben die Möglichkeit, eine Testseite zu drucken. Durch einen Doppelklick auf die Warteschlange gelangen Sie in den Dialog *Warteschlange bearbeiten*, in dessen Dialogblättern Sie unzählige weitere Optionen einstellen können, z. B. den Druckertreiber (bei mehreren Treibern zur Auswahl), die Auflösung bzw. den Qualitätsmodus sowie die Papiergröße.

Beachten Sie, dass Änderungen an der Konfiguration erst wirksam werden, wenn Sie den Button *Übernehmen* anklicken. Anschließend können Sie eine Testseite für die neue Konfiguration ausdrucken (*Test*-Menü). Wenn Sie mehrere Warteschlangen konfiguriert haben, können Sie mit *Standard* festlegen, welche per Default verwendet werden soll (Default-Drucker).

6.1.2 Suse-Drucker-Konfiguration

Zur Konfiguration verwenden Sie das YaST-Modul *Hardware* → *Drucker*. Im günstigsten Fall erkennt YaST2 den Drucker auf Anhieb – andernfalls müssen Sie das Modell aus einer langen Liste auswählen.

YaST richtet bei vielen Druckern gleich mehrere Warteschlangen ein, die sich durch ihre Einstellungen für die Druckqualität und den Farbdruck unterscheiden. Sie können für jede dieser Warteschlangen einen Testdruck durchführen sowie diverse Einstellungen verändern. Leider sind die mehrstufig verschachtelten Dialoge etwas unübersichtlich. In den meisten Fällen ist es aber gar nicht notwendig, sich damit zu plagen: Führen Sie die ersten Tests einfach in der Default-Einstellung durch.

Zum Ändern der Papiergröße wählen Sie zuerst eine Warteschlange aus. Der Button *Bearbeiten* führt in den ersten Optionsdialog. Dort wählen Sie den Eintrag *Einstellungen für den Druckerfilter* aus und klicken wieder auf *Bearbeiten*. Im Dialog *Konfigurationsoptionen* wählen Sie im ersten Listenfeld den Eintrag *Page size* aus; nun können Sie im zweiten Listenfeld aus zahlreichen vordefinierten Papierformaten auswählen.

Wenn Sie mit YaST mehr als einen Drucker konfigurieren, werden die vielen Warteschlangen rasch unübersichtlich. In diesem Fall ist es empfehlenswert, nicht benutzte Warteschlangen wieder zu löschen.

6.1.3 KDE-Drucker-Konfiguration

Das KDE-Kontrollzentrum kann ebenfalls zur Konfiguration von Druckern verwendet werden. Diese Möglichkeit ist dann praktisch, wenn Ihre Distribution kein entsprechendes Konfigurationswerkzeug enthält.

Den Assistenten zur Drucker-Konfiguration starten Sie im Kontrollzentrum-Modul *Angeschlossene Geräte* → *Drucker* durch den Kontextmenüeintrag *Drucker/Klasse hinzufügen*. Dieses Kommando steht nur zur Verfügung, wenn Sie als `root` arbeiten bzw. wenn Sie im Kontrollzentrum in den Systemverwaltermodus gewechselt sind.

> **Hinweis**
>
> Um Konflikte bei der Drucker-Konfiguration zu vermeiden, sollte immer nur *ein* Konfigurationswerkzeug verwendet werden. Wenn Sie also Mandriva, Red Hat, Suse etc. anwenden, dann sollten Sie ausschließlich das zu Ihrer Distribution passende Konfigurationsprogramm verwenden!

Im ersten Dialog geben Sie an, ob Sie einen lokalen Drucker oder einen Netzwerkdrucker konfigurieren möchten. Bei einem lokalen Drucker wählen Sie im zweiten Schritt die Schnittstelle aus, an die der Drucker angeschlossen ist. Im dritten Schritt wählen Sie den Druckerhersteller und das Modell aus. Falls es für das Modell mehrere Treiber gibt, stehen diese im nächsten Dialog zur Auswahl. Damit ist die Basiskonfiguration abgeschlossen. Sie können nun eine Testseite drucken oder mit dem Button *Einstellungen* weitere Optionen wie die Druckqualität oder die Papiergröße einstellen. Oft muss die Papiergröße von der Default-Einstellung *letter* auf A4 umgestellt werden.

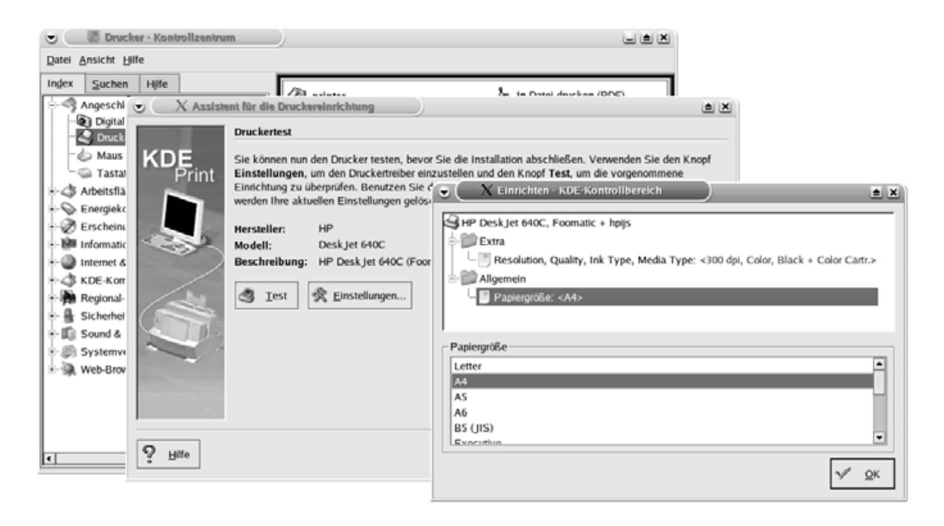


Abbildung 6.1: Drucker-Konfiguration im KDE-Kontrollzentrum

In drei weiteren Dialogen des Assistenten legen Sie nun fest, ob vor oder nach jedem Druckauftrag Trennseiten gedruckt werden sollen, ob es Beschränkungen für die Nutzung des Druckers gibt (max. Dateigröße, Nutzungszeiten, User etc.). Im Regelfall können Sie alle Voreinstellungen unverändert lassen. Zum Abschluss der Konfiguration müssen Sie der neuen Warteschlange einen Namen geben. Optional können Sie im Feld *Adresse* noch angeben, wo sich der Drucker physikalisch befindet. Das ist praktisch, wenn es im Netzwerk mehrere Drucker gibt, die in verschiedenen Räumen bzw. Stockwerken aufgestellt sind.

6.1.4 Drucken

Drucken per Kommando: Von nun an können Sie Text-, Bild-, PostScript- und PDF-Dateien mit dem folgenden Kommando drucken:

`user$` **lpr -P***name* **datei**

Bei `lpr` müssen Sie mit der Option `-P` ohne Leerzeichen den Namen der Warteschlange angeben. Für den Standarddrucker können Sie auf `-P` verzichten.

Durch eine Pipe kann `lpr` auch dazu verwendet werden, die Ausgabe eines anderen Kommandos auszudrucken. Das folgende Kommando druckt die mit `ls` ermittelte Dateiliste auf dem Standarddrucker aus:

`user$` **ls -l *.tex | lpr**

KDE, Gnome und X: Bei allen KDE- und Gnome-Programmen führt *Datei* → *Drucken* zu einem komfortablen Dialog, in dem Sie den Drucker auswählen

können. Der Dialog bietet auch die Möglichkeit, eine PostScript- oder PDF-Datei zu erzeugen.

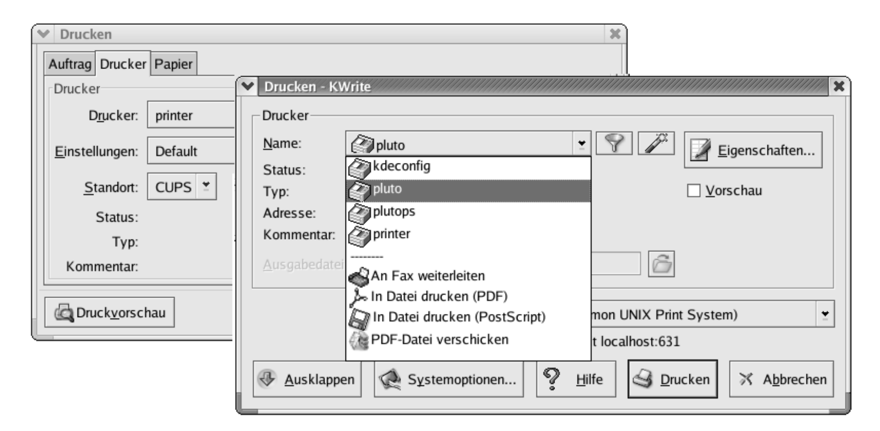


Abbildung 6.2: Druckdialoge von Gnome und KDE

Unter KDE können Sie zudem das Kommando `kprinter` als komfortablen Ersatz für `lpr` verwenden. Das Kommando druckt die Datei bzw. die per Pipe übergebenen Daten aus, zeigt aber vorher den KDE-Druckdialog an.

```
user$ kprinter datei
user$ ls -l | kprinter
```

In den aktuellen Gnome-Versionen haben wir kein Programm gefunden, das eine ähnliche Funktion wie `kprinter` bieten würde. Bei älteren Gnome-Versionen bietet `g-print` diese Funktionalität. Auch das X-Programm `xpp` kann als grafischer Ersatz für `lpr` eingesetzt werden und bietet einen komfortablen Druckerauswahldialog.

Open Office: Auch Open Office bietet einen Druckdialog an, in dem der gewünschte Drucker ausgewählt werden kann. Darüber hinaus gehende Veränderungen an den Druckereinstellungen (Button *Eigenschaften*) werden aber zumeist ignoriert. Zum Drucken im Querformat müssen Sie das Papierformat im Dialogblatt *Format* → *Seite* → *Seite* einstellen.

Mozilla: Mozilla kann per Default nur den Standarddrucker benutzen. Der einfachste Ausweg besteht darin, im Druckerdialog mit dem Button *Eigenschaften* ein externes Programm zur Druckerauswahl anzugeben (z. B. `kprinter`, `g-print` oder `xpp`). Die andere Variante besteht darin, das Zusatzprogramm `xprint` zu installieren.

Sonstige X-Programme: Hier fehlt eine einheitliche Schnittstelle, über die der Drucker angesprochen wird. In den meisten Fällen gibt es aber im Druckdialog zumindest die Möglichkeit, das Druckkommando manuell in der Form `lpr -P`*name* anzugeben.

Druckeradministration: Alle Druckaufträge, die nicht sofort ausgeführt werden können, werden in Warteschlangen zwischengespeichert (je eine pro eingerichteten Drucker). Der Inhalt der Warteschlange kann mit `lpq -P`*name* angesehen werden. Druckjobs, die Sie selbst initiiert haben, können Sie mit `lprm -P`*name* `id` wieder löschen, wobei Sie den Namen der Warteschlange und die ID-Nummer des Jobs angeben müssen. (Die richtige Nummer muss vorher mit `lpq` ermittelt werden.)

Viel komfortabler als `lpq` und `lprm` sind grafische Benutzeroberflächen zur Verwaltung der Druckjobs. Unter KDE können Sie dazu das *Drucker*-Modul des KDE-Kontrollzentrums einsetzen, unter Gnome das Programm `gnome-print-manager`. Außerdem können Sie CUPS mit jedem Webbrowser unter der Adresse *http://localhost:631* administrieren (siehe Seite 269).

6.2 Interna des Druckprozesses

Die gesamte Unix-Druckphilosophie basiert auf PostScript-Druckern. PostScript ist eine Programmiersprache zur Beschreibung von Seiteninhalten. PostScript-Drucker erwarten Druckdaten in diesem Format. Der Vorteil von PostScript gegenüber anderen Formaten besteht darin, dass die Beschreibung ein Vektorformat verwendet und daher in beliebiger Auflösung funktioniert. Dieselbe PostScript-Datei wird umso schärfer ausgedruckt, je besser (je höher auflösend) der Drucker ist. Deswegen spielt PostScript nicht nur unter Unix, sondern vor allem im Druckgewerbe eine dominierende Rolle. Fast alle Unix/Linux-Programme, die Druckfunktionen anbieten, können ausschließlich PostScript-Dateien erzeugen. Es stellt sich also die Frage, wie solche Dateien zum Drucker kommen.

Der Ausdruck ist dann am einfachsten, wenn Sie tatsächlich einen PostScript-Drucker besitzen, der an die parallele Schnittstelle angeschlossen ist, und Sie als `root` arbeiten. In diesem Fall senden Sie die PostScript-Datei einfach mit `cp` an das Device der parallelen Schnittstelle. Linux kennt drei Devices für die parallele Schnittstelle, `/dev/lp0` bis `/dev/lp2`.

```
root#  cp datei.ps /dev/lp0
```

Ist Ihr Drucker an eine serielle Schnittstelle angeschlossen oder handelt es sich um ein USB-Gerät, ändert sich nur der Device-Name (in der Regel existieren vier serielle Schnittstellen (`/dev/ttyS0` bis `/dev/ttyS3`) und etliche für USB:

```
root#  cp datei.ps /dev/ttyS0
root#  cp datei.ps /dev/usb/lp0
```

Spooling-Systeme: Nun wollen normalerweise nicht nur `root` drucken, sondern auch andere Benutzer, möglicherweise nicht nur solche, die lokal am Rechner arbeiten, sondern auch welche, die auf einem anderen Rechner im Netzwerk

arbeiten. Keiner von ihnen möchte sich mit Device-Namen herumärgern (geschweige denn damit, dass der Zugriff auf diese Devices meist nicht erlaubt ist). Aus diesem Grund wurde unter Unix/Linux die ganze Druckerei gleich mit einem Spooling-System konzipiert:

- Es stellt einfach zu bedienende Kommandos zum Drucken zur Verfügung, dank derer beim Ausdruck kein Device-Name, sondern einfach der Druckername angegeben werden muss.
- Es erlaubt je nach Konfiguration allen Benutzern (auch in einem Netzwerk) das Drucken. Deshalb brauchen Sie auf jeden Fall das `loopback`-Netzwerk-Interface.
- Sie ermöglichen es, an einen Rechner mehrere Drucker anzuschließen und zu verwalten.
- Treffen mehrere Druckaufträge gleichzeitig ein, werden die Aufträge in so genannten Warteschlangen (*print queues*) zwischengespeichert, bis der Drucker frei ist.
- Außerdem können Spooling-Systeme diverse Zusatzfunktionen übernehmen, etwa eine Protokollierung, wer wie viel druckt etc.

Unter Linux war lange Zeit das **BSD-LPD**-System üblich (LPD = *line printer daemon*). BSD ist das legendäre Berkely-Unix, in dem der LPD-Code erstmals zum Einsatz kam. Das Drucksystem **LPRng** (LPR next generation) ist dem BSD-LPD sehr ähnlich, aber deutlich sicherer und mit einigen Zusatzfunktionen ausgerüstet. **CUPS** ist das zurzeit modernste System und hat sich in den letzten Jahren als neuer De-facto-Standard durchgesetzt.

Auch wenn sich die drei erwähnten Spooling-Systeme hinter den Kulissen sehr stark voneinander unterscheiden, ist glücklicherweise die Oberfläche einheitlich. Das Kommando zum Drucken einer Datei lautet daher bei allen `lpr` (siehe oben).

Bis hierher muss der Drucker PostScript-kompatibel sein. Das ist aber nur bei sehr wenigen, meist teuren Seitendruckern der Fall. Damit das Drucken einer PostScript-Datei auch sonst funktioniert, ist eine Umwandlung der PostScript-Daten in das jeweilige Druckerformat erforderlich. Dazu kommt das Programm GhostScript (Programmname `gs`) zum Einsatz. Es wandelt die PostScript-Daten seitenweise in Bitmaps um und gibt diese – angereichert mit den Druckbefehlen des jeweiligen Druckers – weiter.

Um den Aufruf von `gs` kümmert sich ein so genannter Filter. Ein Filter ist ein Programm (ein Script), das Eingabedaten verarbeitet und Ausgabedaten liefert. Der Filter für den Druckprozess muss insbesondere die richtigen Parameter an `gs` weitergeben (also den Namen des Druckermodells, die gewünschte Auflösung, die gewünschte Seitengröße etc.).

Nun ist PostScript zwar das Format aller Druckdateien – aber manchmal soll nur eine einfache Text- oder Grafikdatei gedruckt werden. Natürlich können Sie die Textdatei in einen Editor laden, der die Datei dann im PostScript-Format ausdruckt; ebenso können Sie die Grafikdatei mit `xv` oder einem der vielen anderen Grafikprogramme laden und damit (wieder im PostScript-Format) ausdrucken. Aber noch bequemer wäre es, wenn Sie auch für derartige Dateien einfach `lpr datei` ausführen könnten.

Für viele Dateitypen können Sie das tatsächlich. Damit das funktioniert, versucht das Spooling-System, den Typ der zu druckenden Datei zu erkennen. Wenn das gelingt und wenn es sich nicht schon um eine PostScript-Datei handelt, wird die Datei mit geeigneten Programmen in das PostScript-Format umgewandelt. Der Aufruf dieser Konvertierungskommandos erfolgt abermals durch ein Filter-Script.

Und jetzt alles zusammen: Sie haben auf Ihrem Rechner einen Tintenstrahldrucker (kein PostScript) richtig konfiguriert. (Der Druckername sei `hp640`.) Sie möchten die Grafikdatei `omashundertster.png` ausdrucken und führen das folgende Kommando aus:

```
user$  lpr -Php640 omashundertster.png
```

Jetzt laufen die folgenden Operationen ab:

- `lpr` gibt die Datei an das Spooling-System weiter.
- Dieses gibt die Datei an das Filtersystem weiter.
- Der Filter erkennt den Dateityp (PNG) und wandelt die Bitmap in das PostScript-Format um.
- Die PostScript-Daten werden an GhostScript weitergegeben, das die PostScript-Daten in das Format des Druckers umwandelt. Je nach Druckertyp und Auflösung kostet diese Umwandlung eine Menge RAM und temporären Speicherplatz (weil eine riesige Bitmap zur Beschreibung der gesamten Seite gebildet wird).
- Wenn der Drucker nicht mit anderen Druckjobs beschäftigt ist, werden die so aufbereiteten Daten ausgedruckt.

6.3 Spooling-Systeme (CUPS)

Spooling-Systeme sind für die Weiterleitung von Druckaufträgen an den Drucker zuständig. Sie sind auch für die Vor- und Nachverarbeitung der Druckdateien (Input- und Output-Filter) verantwortlich. Manche Druckersysteme delegieren diese Aufgaben an andere Programmkomponenten, nur CUPS vereint sämtliche Funktionen in einem Paket.

Dieser Abschnitt stellt zuerst kurz die alten Systeme BSD-LPD und LPRng vor und widmet sich dann ausführlich dem neuen System CUPS, das mittlerweile von fast allen Distributionen per Default verwendet wird. Nur bei älteren Installationen werden Sie noch oft auf BSD-LPD oder LPRng stoßen.

6.3.1 BSD-LPD

Das BSD-LPD-System (manchmal BSD-LPR oder auch nur LPD genannt) ist gleichsam der Urvater vieler Spooling-Systeme. Für die Verwaltung des Spoolers ist der Drucker-Dämon `lpd` verantwortlich, der während des Init-V-Prozesses gestartet wird. Beim Start dieses Hintergrundprozesses wird die Datei `/etc/printcap` ausgewertet. Diese Datei enthält in einer leider ziemlich unübersichtlichen Syntax die Daten zur Konfiguration des Druckers (oder der Drucker, falls mehrere angeschlossen sind). Eine Minimalkonfiguration eines PostScript-Druckers ohne Filter kann folgendermaßen aussehen:

```
# /etc/printcap
# local
lp|aliasname:\
  :lp=/dev/lp0:\
  :sd=/usr/spool/lp0:\
  :mx#0:\
  :sh:
# remote
lp2:\
  :lp=:\
  :rm=remotehost:\
  :rp=remoteprinter:\
  :sd=/usr/spool/lp1:\
  :mx#0:\
  :sh:
```

Der Drucker kann wahlweise mit dem Namen `lp` oder `aliasname` angesprochen werden. (Mit `|` können weitere Alias-Namen angegeben werden. Der Name `lp` hat bei BSD-LPD eine besondere Bedeutung und bezeichnet den Defaultdrucker.)

`/etc/printcap`-Parameter können auch in einer langen Zeile (ohne `\`-Zeichen) angegeben werden. Bei einer Aufteilung über mehrere Zeilen (wie im obigen Beispiel) müssen die einzelnen Einträge beidseitig durch Doppelpunkte eingeklammert sein, sonst kommt `lpd` durch die Leerzeichen der Einrückung in Schwierigkeiten. Die folgende Liste zählt die wichtigsten `printcap`-Schlüsselwörter auf. (Ausführlichere Informationen liefert `man printcap`.)

- `af=` bezeichnet die Datei, in der alle Druckjobs protokolliert werden.

- `if=` gibt den Programm- bzw. Script-Namen des Filters an. (Der Filter kümmert sich um die Umwandlung der Ausgangsdaten in das PostScript-Format. Wenn kein PostScript-Drucker verwendet wird, werden anschließend die PostScript-Daten in das Format des Druckers umgewandelt. Der Filter kann schließlich auch die Protokollierung der Druckjobs durchführen.)
- `lf=` bezeichnet die Datei für die Fehlermeldungen.
- `lp=` stellt das Drucker-Device ein.
- `mx#0` bewirkt, dass die Größe von Druckdateien unbegrenzt ist.
- `sd=` stellt das Verzeichnis ein, in dem Spool-Dateien zwischengespeichert werden sollen.
- `sh` bewirkt, dass `lpd` auf den Ausdruck einer Titelseite vor jedem Ausdruck verzichtet.
- `tr=` gibt eine Zeichenkette an, die zum Abschluss an den Drucker gesandt wird.
- `rm=` gibt den Print-Server beim Drucken im Netz an (ggf. auch den Namen oder die IP-Adresse des netzwerkfähigen Druckers).
- `rp=` gibt den Namen des Remote-Printers an (default: lp1).

Wenn das BSD-LPD-System als Drucker-Server arbeiten soll, also auch Druckjobs von anderen Rechnern annehmen soll, müssen deren Rechnernamen in `/etc/hosts.lpd` aufgezählt werden.

6.3.2 LPRng

LPRng (*Line Printer next generation*) ist eine komplette Neuimplementierung des BSD-LPD-Systems. LPRng ist weitestgehend kompatibel zu BSD-LPD, ist aber wesentlich sicherer, bietet mehr Möglichkeiten zur Einstellung der Zugriffsrechte im Netzwerk und erlaubt die Bildung von Druckerpools, um Druckaufträge automatisch an den ersten freien Drucker zu leiten. Weitere Informationen zu LPRng finden Sie unter *http://www.lprng.com*.

Die Konfiguration von LPRng erfolgt durch die folgenden drei Dateien, zu denen es jeweils ausführliche `man`-Seiten gibt `/etc/lpd.conf` für die globale LPRng-Konfiguration, `/etc/printcap` zur Drucker-Konfiguration (ähnlich LPR-BSD) und `/etc/lpd.perms` für die Zugriffsrechte (insbesondere für Netzwerkdrucker).

6.3.3 CUPS

CUPS steht für *Common Unix Printing System*. Das CUPS-Projekt wurde mit dem Ziel gegründet, die Druckerverwaltung unter Unix/Linux vollkommen neu zu organisieren. Zu den wesentlichsten Vorteilen von CUPS zählen:

- CUPS bietet eine weitaus bessere Integration aller druckerrelevanten Funktionen. CUPS kümmert sich nicht nur um das Spooling, sondern stellt auch das Filtersystem zur Verfügung, bietet Hilfsmittel zur Drucker-Konfiguration und zur Einstellung von Druckerparametern etc. CUPS und GhostScript stellen insofern eine Komplettlösung dar (während andere Drucksysteme aus viel mehr Einzelteilen bestehen, deren Zusammenarbeit oft Probleme bereitet).
- Der Input-Filter (Dokument-Filter) basiert auf MIME-Dateien, was die Konfiguration wesentlich vereinfacht.
- Zur Drucker-Konfiguration können die weit verbreiteten PPD-Dateien verwendet werden (*PostScript Printer Description*). Diese Dateien eignen sich auch zur Beschreibung von Nicht-PostScript-Druckern.
- Die Druckeradministration kann auch über das Netz erfolgen. CUPS enthält dazu einen eigenen HTTP-Server. Laden Sie mit Ihrem Webbrowser die Seite *http://localhost:631*.
- Die Netzwerk-Konfiguration von CUPS (siehe auch Seite 270) orientiert sich stark an der von Apache.
- CUPS unterstützt das neue Internet-Printing-Protocol (IPP). Ob sich dieses Protokoll bei Druckerherstellern und/oder bei der Drucker-Software durchsetzt, bleibt aber noch abzuwarten. Detaillierte Informationen zu IPP finden Sie unter *http://www.pwg.org/ipp/*.
- Wie bei LPRng können Klassen eingerichtet werden, um Drucker-Pools zu konfigurieren.
- Dank CUPS ist es möglich, einen raschen Überblick über alle im Netzwerk verfügbaren Drucker zu erhalten (*printer browsing*).

Zu CUPS steht eine umfassende Dokumentation zur Verfügung (wahrscheinlich mehr, als Sie je lesen werden), lokal unter *http://localhost:631/* und im Internet bei *http://www.cups.org/*.

Normalerweise sollten Sie zur Konfiguration das Konfigurationsprogramm Ihrer Distribution oder das KDE-Kontrollzentrum einsetzen. Manuelle Änderungen an der Konfiguration sind nur empfehlenswert, wenn Sie die gesamte CUPS-Dokumentation gelesen und verstanden haben.

Bei CUPS heißt der im Hintergrund laufende Drucker-Dämon `cupsd`. Das Programm wird während des Init-V-Prozesses gestartet (üblicherweise durch

`/etc/init.d/cups`). Damit CUPS auch Druckjobs von anderen Rechnern im lokalen Netz annehmen kann, die selbst nicht CUPS als Drucksystem verwenden und daher serverseitig den herkömmlichen Dämon `lpd` des BSD-LPD-Systems erwarten, gibt es den Mini-Dämon `cups-lpd`. Dieses Programm wird nur bei Bedarf durch `xinetd/inetd` gestartet. Die erforderliche Konfiguration für `xinetd` ist in `man cups-lpd` beschrieben und sollte von der Distribution automatisch eingerichtet werden.

Beim BSD-LPD-System und bei LPRng erfolgt beinahe die gesamte Drucker-Konfiguration durch die Datei `/etc/printcap`. Bei CUPS spielt diese Datei dagegen so gut wie keine Rolle mehr. Sie steht zwar aus Kompatibilitätsgründen noch immer zur Verfügung, enthält aber nur eine Liste aller bekannten Warteschlangen (ohne irgendwelche weiteren Parameter).

Die eigentliche CUPS-Konfiguration erfolgt durch die Dateien des Verzeichnisses `/etc/cups`. Die folgende Tabelle zählt die wichtigsten Dateien auf:

`classes.conf`	Definition aller Klassen
`cupsd.conf`	Zentrale CUPS-Konfigurationsdatei
`lpoptions`	Veränderungen gegenüber der Default-Konfiguration
`mime.convs`	Zu verwendende Filter für verschiedene Dateitypen
`mime.types`	Dateitypen für die PostScript-Konvertierung
`ppds.dat`	PPD-Datenbank
`printers.conf`	Definition aller Drucker
`ppd/name.ppd`	Konfiguration für die Warteschlange *name*

PPD-Dateien (PostScript Printer Definition)

Für CUPS sieht jeder Drucker wie ein PostScript-Drucker aus. Druckerspezifische Details wie die Größe des nicht bedruckbaren Seitenrands, die Auflösung, Kommandos für bestimmte Zusatzfunktionen (z. B. Papiereinzug), Besonderheiten (Duplex-Druck) etc. werden in so genannten PPD-Dateien gespeichert (*PostScript Printer Definition*). Das PPD-Format wurde von Adobe definiert und kommt auch unter Windows und auf Apple-Rechnern zum Einsatz.

Da natürlich nicht jeder Drucker tatsächlich ein PostScript-Drucker ist, enthalten CUPS-PPD-Dateien in Form von Kommentaren auch das erforderliche GhostScript-Kommando inklusive aller Optionen, damit `gs` die PostScript-Datei in das Format des Druckers umwandeln kann. Die folgenden Zeilen zeigen einige Auszüge aus einer PPD-Datei für den Tintenstrahldrucker HP DeskJet 640:

```
*PPD-Adobe: "4.3"
...
*Manufacturer:  "HP"
*Product:       "(DeskJet 640C)"
*cupsVersion:   1.0
*cupsManualCopies: True
*cupsModelNumber:  2
```

```
*cupsFilter:    "application/vnd.cups-postscript 0 foomatic-rip"
*%pprRIP:        foomatic-rip other
...
*FoomaticIDs: HP-DeskJet_640C hpijs
*FoomaticRIPCommandLine: "gs -q -dBATCH -dPARANOIDSAFER -dQUIET &&
  -dNOPAUSE -sDEVICE=ijs -sIjsServer=hpijs%A%B%C &&
  -dIjSuseOutputFD%Z -sOutputFile=- -"
*End
...
```

`ppds.dat` enthält eine Datenbank mit allen für CUPS bekannten PPD-Einträgen. Diese Datenbank ist die Grundlage zur Konfiguration eines neuen Druckers. Während der Konfiguration eines Druckers wird der passende Eintrag aus der Datenbank extrahiert und in `/etc/cups/ppd/name.ppd` gespeichert.

cupsomatic: Das Druck-Puzzlespiel wird durch den so genannten `cupsomatic`-Filter abgeschlossen. Beim Ausdruck einer Datei kümmert sich CUPS zuerst um die Umwandlung der Druckdatei in das PostScript-Format. Anschließend wird das Script `cupsomatic` aufgerufen. Es extrahiert aus der `name.ppd`-Datei für den angegebenen Drucker die GhostScript-Parameter, ruft damit `gs` auf und wandelt so die PostScript-Daten in das Format des jeweiligen Druckers um. Die resultierenden Daten werden dann an das Drucker-Device gesendet.

6.3.4 CUPS-Kommandos

Üblicherweise werden Sie sowohl zum Drucken als auch zur CUPS-Administration die Werkzeuge von KDE oder Gnome verwenden, die gut mit CUPS abgestimmt sind (KDE-Kontrollzentrum oder `gnome-print-manager`). Auch das CUPS-Webinterface hilft bei der Administration (siehe Seite 269). Manchmal ist es aber praktisch, zur Administration ganz gewöhnliche Kommandos zu verwenden. Die wichtigsten Kommandos werden hier kurz beschrieben. Die Kommandos `lpr`, `lpq`, `lprm` und `lpc` stehen nicht nur bei CUPS, sondern auch bei BSD-LPD und LPRng zur Verfügung. Das ist gewissermaßen ein gemeinsamer Nenner aller Spooling-Systeme. Beachten Sie aber, dass es Unterschiede bei den unterstützten Optionen gibt.

`lpr` `[optionen] [datei]`

Wie bereits mehrfach erwähnt wurde, sendet das Kommando `lpr datei` die Datei an den Standarddrucker. Die Ausgabe anderer Programme kann in `lpr` „gepiped" werden (z. B. `ls -l | lpr`). Statt `lpr` kann auch das Kommando `lp` verwendet werden (Syntax siehe `man 1 lp`).

`-l` umgeht das sonst übliche Filtersystem und sendet die Daten unverändert an den Drucker. Die Option ist nur sinnvoll, wenn eine Druckdatei bereits im druckerspezifischen Format vorliegt.

`-o options`	übergibt diverse Zusatzparameter, z. B. `-o media=A4` oder `-o page-ranges=23-27,29,31`. (Weitere Beispiele finden Sie im *CUPS Software Users Manual*.)
`-Pname`	verwendet die Warteschlange *name* statt des Standarddruckers. Beachten Sie, dass dem „P“ kein Leerzeichen folgt!

lpc

Das Kommando `lpc` gestattet eine feinere Kontrolle über den Druckvorgang. Nach dem Start dieses Kommandos befinden Sie sich in einer interaktiven Arbeitsumgebung, in der Sie Kommandos wie `status`, `help` etc. ausführen können. Mit `topq` können Sie einen Druckjob in der Warteliste nach oben befördern. Als Parameter müssen Sie den Druckernamen und die Jobnummer angeben. Ein Teil der Kommandos in `lpc` (so auch `topq`) darf nur von `root` ausgeführt werden. `exit`, `bye` oder `quit` beenden `lpc`.

lpq `[optionen]`

Alle an den Drucker gesandten Daten werden im Verzeichnis `/var/spool/cups/*` zwischengespeichert, bis der Ausdruck abgeschlossen ist. Das Kommando `lpq` liefert eine Liste aller zwischengespeicherten Dateien bzw. Druckaufträge mit Jobnummer und Dateigröße. Diese Jobnummer können Sie als Parameter von `lprm` angeben, um eine Datei aus dem Drucker-Spooler zu löschen.

`-a`	zeigt die Druckjobs aller Warteschlangen an.
`-Pname`	zeigt die Druckjobs der angegebenen Warteschlange *name* an. Beachten Sie, dass dem „P“ kein Leerzeichen folgt!

Alle Spool-Daten werden auf der Festplatte gespeichert. `lpd` stellt nach einem Neustart fest, dass es noch nicht ausgedruckte Dateien gibt und wird weiterhin versuchen, die Daten an den Drucker zu übertragen.

lprm `[optionen] [id]`

`lprm` bricht den aktuellen bzw. den durch die ID-Nummer angegebenen Druckjob ab.

`-Pname`	gibt die Warteschlange *name* an. Beachten Sie, dass dem „P“ kein Leerzeichen folgt!

lpstat [optionen]

Das Kommando `lpstat` zeigt Informationen über Klassen, Drucker und deren Druckjobs an. `lpstat` gilt nicht nur für lokale Drucker. Das Programm zeigt automatisch Informationen über alle im Netzwerk zugänglichen CUPS-Drucker an.

`-a` zeigt für alle Drucker an, ob sie bereit sind, Druckaufträge entgegenzunehmen.

`-c` zeigt alle Klassen an.

`-d` zeigt den Default-Drucker an.

`-s` zeigt eine Statusübersicht an (Default-Drucker, Liste aller Klassen und Drucker samt Device-Namen etc.).

`-t` zeigt alle verfügbaren Informationen an.

`-v` zeigt alle Drucker an (mit Informationen darüber, wo im Netzwerk sich diese Drucker befinden).

lpinfo [optionen]

Das Kommando zeigt alle bekannten Druck-Devices (`-v`) bzw. alle bekannten Druckertreiber an (`-m`). Noch ausführlichere Informationen erhalten Sie mit der zusätzlichen Option `-l`.

lpadmin [optionen]

Mit diesem Kommando können CUPS-Drucker und -Klassen eingerichtet und wieder gelöscht werden. Außerdem können die Zugriffsrechte auf den Drucker (speziell für den Netzwerkbetrieb) eingestellt werden.

lpoptions [optionen]

`lpoptions` zeigt Druckoptionen an bzw. verändert sie. Wenn das Kommando nicht für den Default-Drucker gelten soll, bestimmt die Option `-p name` den Drucker.

```
user$ lpoptions -p normal
  job-sheets=none,none ImageableArea=A4 PageRegion=A4
  PageSize=A4 PaperDimension=A4 PrintoutMode=
```

lphelp name

`lphelp` zeigt Detailinformationen zur angegebenen Warteschlange an.

```
user$ lphelp normal
HP DeskJet 640C
   Colour printer

   Printer-specific options

   Page Size:  -o PageSize=<choice>
      <choice> can be one of the following:
      Custom  (Custom Size, size: 0.00x0.00in)
      A4  (A4, size: 8.26x11.69in, default)
      ...

   Printout Mode:  -o PrintoutMode=<choice>
      <choice> can be one of the following:
      Draft  (Draft)
      Draft.Gray  (Draft Grayscale)
      ...
...
```

6.3.5 CUPS-Webadministration

Die Administration von CUPS kann auch über das Internet erfolgen. CUPS enthält dazu einen eingebauten HTTP-Server, der per Default über den Port 631 des *Internet Printing Protocols* IPP zugänglich ist. Zum Ausprobieren betrachten Sie mit Ihrem Webbrowser die Seite `http://localhost:631`.

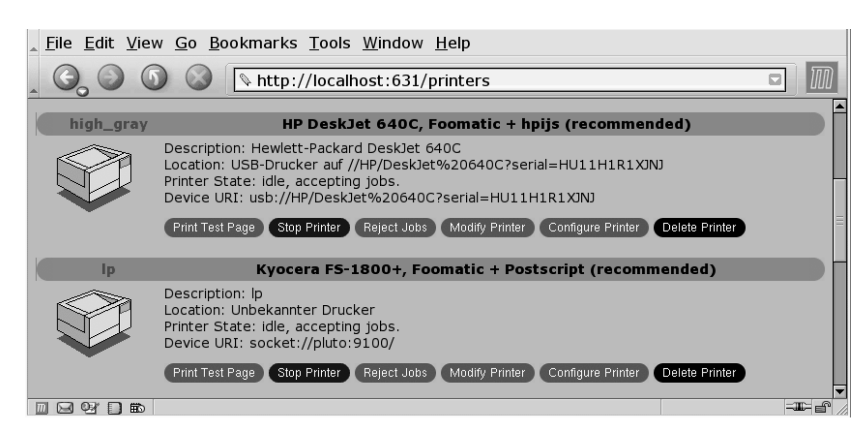


Abbildung 6.3: CUPS-Webadministration

Es hängt von den Einstellungen in `/etc/cups/cupsd.conf` ab, ob die Webadministration überhaupt möglich ist, ob Sie nur vom lokalen Rechner oder auch im Netzwerk genutzt werden kann und ob dazu ein Login erforderlich ist.

6.4 Netzwerkdrucker anbieten (Server)

In einem Netzwerk besteht oft der Wunsch, den lokal eingerichteten Drucker auch anderen Teilnehmern im Netzwerk zur Verfügung zu stellen. Das ist an sich kein Problem, weil CUPS als Drucker-Server konzipiert ist: Das Programm ist also darauf vorbereitet, Druckjobs von anderen Rechnern anzunehmen. Im Folgenden wird vorausgesetzt, dass die Einrichtung des lokalen Druckers bereits gelungen ist, d. h., dass der Drucker vom lokalen Rechner aus genutzt werden kann.

Andere Rechner können diesen Drucker nur dann nutzen, wenn Ihr CUPS-Server dies zulässt (Zugriffskontrolle). Damit alle Rechner im lokalen Netzwerk wissen, welche Drucker überhaupt zur Verfügung stehen, müssen außerdem die Browsing-Funktionen des CUPS-Servers aktiviert werden. (Damit die Drucker auch in einem Windows-Netzwerk verwendet werden können, muss außer CUPS auch Samba konfiguriert werden.)

Browsing-Funktion: Dank der Browsing-Funktion können alle im lokalen Netzwerk verbundenen Rechner sich mit CUPS gegenseitig mitteilen, welche Drucker dort angeschlossen sind. Bei vielen Distributionen ist die Browsing-Funktion per Default allerdings nicht aktiv.

Zur Aktivierung müssen Sie die beiden folgenden Zeilen in /etc/cups/cupsd.conf einfügen bzw. ändern. Die *BrowseAddress*-Einstellung bewirkt, dass die CUPS-Informationen an alle lokalen Netzwerkschnittstellen gesendet werden (*broadcast*), nicht aber an Internet-Schnittstellen (PPP etc.). Alternativ kann auch eine bestimmte Netzwerkschnittstelle mit BrowseAddress @IF(eth0) angegeben werden.

```
# in /etc/cups/cupsd.conf
Browsing On
BrowseAddress @LOCAL
```

cupsd.conf sieht eine Reihe weiterer *Browse*-Schlüsselwörter vor. Beispielsweise steuern *BrowseAllow* und *BrowseDeny*, von welchen Rechnern CUPS-Informationen *empfangen* werden. Per Default gibt es keine Empfangseinschränkungen und es ist selten notwendig oder sinnvoll, diese oder die anderen *BrowseXxx*-Einstellungen zu ändern. Damit die Änderungen an cupsd.conf gültig werden, müssen Sie CUPS mittels /etc/init.d/cups restart neu starten.

Zugriffssteuerung: Der Zugriff auf verschiedene CUPS-Komponenten wird durch *¡Location xxx¿*-Einträge in /etc/cups/cupsd.conf gesteuert. Die Default-Konfiguration ist stark von der jeweiligen Distribution abhängig. Die folgende Tabelle beschreibt die Bedeutung der wichtigsten Einträge:

<Location />	Default-Einstellungen für alle Komponenten
<Location /admin>	Administrationsrechte (Webadministration)

`<Location /jobs>`	Kontrolle über Druck-Jobs
`<Location /printers>`	Zugriff auf alle Drucker (Default-Einstellung)
`<Location /printers/name>`	Zugriff auf einen bestimmten Drucker

Bei der folgenden Beispielkonfiguration ist grundsätzlich nur vom Rechner *localhost* ein Zugriff auf alle CUPS-Funktionen möglich. Aus dem lokalen Netz ist außerdem der Zugriff auf alle Drucker, auf die Druckjobkontrolle und auf die Administrationsfunktionen erlaubt. Für die Jobkontrolle ist allerdings ein Login erforderlich (*AuthType Basic* und *AuthClass User*), für die Administration sogar ein *root*-Login (*AuthClass System*). Die Zeile *Allow From 127.0.0.2* ist nur bei Suse erforderlich, wo *localhost* auch diese IP-Adresse haben kann. (Eine ausführliche Beschreibung aller Schlüsselwörter zur Steuerung der CUPS-Zugriffsrechte finden Sie im *CUPS Software Administrators Manual* unter der Adresse *http://localhost:631/*.).

```
# Steuerung der Zugriffsrechte in /etc/cups/cupsd.conf
<Location />
  Order Deny,Allow
  Deny From All
  Allow From 127.0.0.1
  Allow From 127.0.0.2
</Location>

<Location /printers>
  Order Deny,Allow
  Deny From All
  Allow From 127.0.0.1
  Allow From 127.0.0.2
  Allow From 192.168.0.*
  Allow From *.sol
</Location>

<Location /jobs>
  AuthType Basic
  AuthClass User
  Order Deny,Allow
  Deny From All
  Allow From 127.0.0.1
  Allow From 127.0.0.2
  Allow From 192.168.0.*
  Allow From *.sol
</Location>

<Location /admin>
  AuthType Basic
  AuthClass System
  Order Deny,Allow
```

```
  Deny From All
  Allow From 127.0.0.1
  Allow From 127.0.0.2
  Allow From 192.168.0.*
  Allow From *.sol
</Location>
```

Hinweis

In den CUPS-Konfigurationsdateien gelten besondere Regeln für Kommentare: Kommentare werden zwar wie üblich mit dem Zeichen # eingeleitet, dieses Zeichen muss aber das erste Zeichen der Zeile sein! Eingerückte Kommentare oder Zeilen, an die ein Kommentar angehängt wird, sind nicht zulässig! Unter Suse sollten Sie generell auf Einrückungen in `cups.conf` verzichten – sonst kann es passieren, dass YaST Ihre Konfigurationsdatei zerstört!
Bei Suse ist ausserdem CUPS mit einem zusätzlichen Patch so kompiliert, dass zur Zugriffskontrolle auch der TCP-Wrapper zum Einsatz kommt. Deswegen funktioniert CUPS nur dann, wenn dies in `/etc/hosts.deny` nicht verboten oder in `/etc/hosts.allow` explizit erlaubt ist. Per Default ist das der Fall. Wenn Sie die beiden Dateien aber selbst verändern, müssen Sie unbedingt auch CUPS berücksichtigen (Eintrag `cupsd`).
Bei Red Hat bzw. Fedora ist `cupsd.conf` so voreingestellt, dass grundsätzlich kein Netzwerkzugriff auf die konfigurierten Drucker möglich ist. Statt `cupsd.conf` manuell zu ändern, können Sie den Netzwerkzugriff auf einen Drucker ganz bequem in `redhat-config-printer` durch das Kommando *Aktion* → *Sharing* freigeben.

6.4.1 BSD-LPD-Kompatibilität

Viele Drucker-Clients setzen voraus, dass als Drucker-Server ein BSD-LPD-kompatibles System läuft. CUPS erfüllt diese Voraussetzung eigentlich nicht, es gibt aber glücklicherweise das Programm `cups-lpd`, das die Kompatibilität herstellt.

`cups-lpd` wird durch `xinetd` gesteuert. `xinetd` ist für den Start diverser Netzwerkdienste zuständig. Entscheidend sind drei Dinge:

- Die Datei `/etc/xinet.d/cups-lpd` muss existieren.
- Diese Datei darf nicht die Einstellung `disable = yes` enthalten. Aus Sicherheitsgründen ist das aber zumeist der Fall. Sie müssen diese Zeile also entfernen bzw. in `disable = no` ändern. Bei Red Hat bzw. Fedora können Sie das bequem mit `redhat-config-printer` erledigen (Menükommando *Aktion* → *Sharing*, Dialogblatt *Allgemein*, Option *LPD Protokoll aktivieren*).

- Die Nutzung von `cups-lpd` muss in `/etc/hosts.allow` erlaubt werden bzw. darf nicht von `/etc/hosts.deny` verboten werden. Beachten Sie, dass die übliche Zugriffskontrolle durch `cupsd.conf` für den LPD-kompatiblen Druck nicht zur Anwendung kommt!

6.5 Netzwerkdrucker nutzen (Client)

Nun geht es um den Ausdruck eines Dokuments auf einem Drucker, der nicht lokal mit dem Rechner verbunden, sondern nur über das Netzwerk zugänglich ist. Der Drucker muss über einen anderen Computer mit dem Netzwerk verbunden sein, wobei es die folgenden Varianten gibt:

- IPP-Drucker (Verwaltung durch Linux/Unix mit CUPS, Internet Printing Protocol)
- Linux/Unix-Drucker (Verwaltung durch Linux/Unix, LPD-Protokoll)
- Windows-Drucker (Verwaltung durch einen Windows-Rechner oder Samba-Server)
- Novell-Drucker (Verwaltung durch einen Novell-Networks-Rechner)

Darüber hinaus bieten die meisten Druckerhersteller in der gehobenen Preisklasse auch Netzwerkdrucker an, die einen integrierten Drucker-Server aufweisen. Sie werden direkt an das lokale Netzwerk angeschlossen und bekommen einen eigenen Netzwerk-Namen. Fast alle derartigen Geräte können so verwendet werden, als würde es sich um Linux/Unix-Drucker (LPD), Windows-Drucker, Novell-Drucker oder IPP-Drucker handeln. Manche Geräte bieten darüber hinaus weitere Protokolle an:

- Socket-API (z. B. JetDirect von HP) am IP-Port 9100
- AppSocket (z. B. Tektronix)
- Herstellerspezifische Protokolle

CUPS-zu-CUPS-Druck: Am einfachsten funktioniert das Drucken im Netzwerk, wenn auf beiden Seiten CUPS bzw. das Protokoll IPP verwendet wird. Derartige Drucker sind ohne weitere Konfigurationsarbeiten auf der Client-Seite sichtbar und können sofort verwendet werden. CUPS trägt alle im Netzwerk sichtbaren Drucker automatisch in `/etc/printcap` ein. `lpstat -v` liefert eine Liste aller verfügbaren Drucker. Das folgende Kommando wurde am Rechner `uranus` ausgeführt. Dort ist lokal nur ein Drucker mit dem Namen *hp640* angeschlossen. Außerdem sind am Rechner `mars` aber noch zwei Drucker mit den Namen `lp` und `lp1` verfügbar.

```
user@uranus$ lpstat -v
device for hp640: usb:/dev/usb/lp0
device for lp: ipp://mars/printers/lp
device for lp1: ipp://mars/printers/lp1
```

Sie können alle drei Drucker sofort mit `lpr -P`*name* benutzen. Falls mehrere Drucker im Netzwerk denselben Namen haben, müssen deren Namen in der Form *druckername@hostname* angegeben werden, also z. B. `lpr -Plp@jupiter`.

Konfiguration für andere Netzwerkdrucker: Wenn die externen Netzwerkdrucker nicht CUPS-kompatibel sind, muss der Drucker client-seitig konfiguriert werden. Dazu können die üblichen Konfigurationsprogramme eingesetzt werden (`redhat-config-printer`, YaST, KDE-Kontrollzentrum, CUPS-Webadministration etc.). Bei Suse haben die verschiedenen Varianten bisweilen merkwürdige Namen: um einen LPD-Drucker zu nutzen, müssen Sie eine *LPD-Vorfilter- und Weiterleitungs-Warteschlange* konfigurieren. Lassen Sie sich von derartigen Wortungetümen nicht irritieren, die eigentliche Konfiguration ist unkompliziert! Bei jedem Netzwerkdrucker muss die IP-Adresse oder der Netzwerkname angegeben werden, bei Windows-Druckern eventuell auch ein Benutzername und ein dazu gehörendes Passwort.

Detailinformationen für die Netzwerkdrucker – also Protokoll, Login-Name etc. – werden in der Datei `/etc/cups/printers.conf` gespeichert. Das entscheidende Schlüsselwort ist dabei *DeviceURI*, das das Protokoll und die Netzwerkadresse angibt. Die folgenden Zeilen geben einige Beispiele:

```
# in /etc/cups/printers.conf
# lokale Drucker (parallel, seriell, USB)
DeviceURI parallel:/dev/lp0
DeviceURI serial:/dev/ttyS0?baud=115200
DeviceURI usb:/dev/usb/lp0

# LPD-Netzwerkdrucker
DeviceURI lpd://saturn/hp3

# Drucker mit Socket-Protokoll (z.B. HP JetDirect)
DeviceURI socket://pluto.sol:9100

# Windows-Drucker
#    Server-Name:              saturn
#    Workgroup-Name:           sol
#    Druckername (Share-Name): myprinter
#    Benutzername:             user
#    Passwort:                 xxx
DeviceURI smb://saturn/myprinter
DeviceURI smb://SOL/saturn/myprinter
DeviceURI smb://user:xxx@saturn/myprinter
DeviceURI smb://user:xxx@SOL/saturn/myprinter
```

Der Vollständigkeit halber folgt hier noch die Syntax, wie der Device-Name eines externen CUPS-Druckers aussehen würde. (Wie bereits oben erklärt wurde, ist es im Regelfall nicht erforderlich, dazu eine Warteschlange einzurichten!)

```
# IPP/CUPS-Drucker
DeviceURI ipp://saturn.sol:631/printers/lp3
```

Windows-Drucker nutzen: Bevor Sie einen Windows-Drucker verwenden können, müssen Sie die Samba-Client-Tools installieren (bzw. bei Distributionen, die nicht zwischen Client- und Server-Tools differenzieren, das komplette Samba-Paket).

6.6 PostScript- und PDF-Tools

6.6.1 PostScript- und PDF-Viewer

ghostview: Zum Betrachten von PostScript-Dateien stehen verschiedene Programme zur Auswahl, deren Urvater `ghostview` ist – eingentlich nur eine Benutzeroberfläche zu dem bereits vorgestellten Programm `gs` (GhostScript). `ghostview` kommt sowohl mit einzelnen PostScript-Grafiken als auch mit Dokumenten zurecht, die Hunderte von Seiten umfassen. Wenn das Dokument den PostScript-Normen entspricht, kann beliebig zwischen den Seitennummern hin- und hergesprungen werden. Andernfalls ist nur ein sequenzielles Lesen der Datei möglich (also eine Seite nach der anderen und ohne die Möglichkeit, Seiten zu überspringen oder zurückzublättern).

Wenn Sie Wert auf eine hohe Darstellungsqualität legen, sollten Sie die Anti-Aliasing-Funktion aktivieren. Das bedeutet, dass das PostScript-Abbild zuerst in einer höheren Auflösung erstellt und dann verkleinert wird. Dazu erweitern Sie die Datei `~/.Xdefaults` bzw. `~/.Xresources` um die folgende `arguments`-Einstellung (die hier nur aus Platzgründen auf zwei Zeilen verteilt ist):

```
Ghostview*arguments: -sDEVICE=x11 -dTextAlphaBits=4
  -dGraphicsAlphaBits=2 -dMaxBitmap=10000000
```

kghostview, gv, ggv: `kghostview`, `gv` und `ggv` sind KDE- und Gnome-Varianten zu `ghostview`. Alle drei Programme zeichnen sich durch eine wesentlich komfortablere Bedienung aus. Ein weiterer Pluspunkt ist, dass die Programme auch mit PDF-Dateien zurechtkommen. Welches der Programme Sie bevorzugen werden, ist letztlich eine Geschmacksfrage.

Acrobat Reader und xpdf: PDF-Dateien können mit den drei gerade erwähnten PostScript-Viewern, mit den Programmen `xpdf`, `gpdf` (ab Gnome 2.4), `kpdf` (ab KDE 3.2) sowie mit dem kommerziellen, aber kostenlos verfügbaren Acrobat Reader von Adobe angezeigt werden (Kommando `acroread`).

Es hängt vom Ursprung der PDF-Datei ab, welches Programm dabei die beste Darstellungsqualität erzielt – aber in vielen Fällen ist das noch immer der Acrobat Reader. Das Programm hat einen weiteren Vorteil: Es kann bei entsprechend gestalteten PDF-Dokumenten Zusatzinformationen anzeigen, die für PostScript-Dateien nicht zur Verfügung stehen, z. B. ein aufklappbares Inhaltsverzeichnis, ein Stichwortverzeichnis sowie Thumbnails. Leider fehlt `acroread` bei vielen Distributionen. Sie finden das Programm gegebenenfalls unter *http://www.adobe.com/products/acrobat/alternate.html*.

6.6.2 Konvertierung Text → PostScript

Bevor einfache Textdateien gedruckt werden können, müssen sie in das PostScript-Format umgewandelt werden. Normalerweise kümmert sich der Input-Filter des Spooling-Systems automatisch um diese Aufgabe. Haben Sie besondere Wünsche zur Formatierung des Ausdrucks, empfiehlt sich eine manuelle Konvertierung und ein anschließender Ausdruck der PostScript-Datei. Dazu stellen wir Ihnen zwei Kommandos vor: `a2ps` und `enscript`.

```
a2ps [optionen] textdatei -o psdatei
```

`a2ps` steht für *Any to PostScript*. Tatsächlich kommt das Programm mit zahlreichen Dateitypen zurecht, auch wenn an dieser Stelle nur die Textkonvertierung beschrieben wird.

In der Default-Einstellung formatiert `a2ps` den Text in einer zweispaltigen Seite im Querformat. Die beiden Spalten werden automatisch mit Rahmen und Überschriften versehen. Wenn es sich bei dem Text um Programmcode handelt und `a2ps` die Programmiersprache erkennt, wird automatisch eine Syntaxhervorhebung durchgeführt (Schlüsselwörter fett, Kommentare kursiv etc.).

Im Folgenden werden nur solche Optionen vorgestellt, die nicht ohnedies als Default-Einstellung gelten:

`-1`	Einspaltiger Druck (statt zweispaltig Landscape)
`-f` *n*	Stellt die Schriftgröße auf *n* Punkt ein. Damit ändert sich automatisch auch der Zeilenabstand und die Anzahl der Zeilen pro Seite (*n* ist Fließkommazahl).
`-R`	Längsformat (Portrait) statt Querformat (Landscape)
`-B`	Keinen Rahmen um den Text drucken
`--delegate=no`	Datei nicht durch ein anderes Programm verarbeiten. (Per Default versucht `a2ps` bei einigen Dateitypen eine automatische Weiterverarbeitung, bei `*.tex`-Dateien z. B. durch `texi2dvi`.)

`--highlight-level =none`	Keine Syntaxhervorhebung durchführen
`-o dateiname`	speichert den resultierenden PostScript-Code in der angegebenen Datei. Andernfalls überträgt `a2ps` den PostScript-Code automatisch an den Standarddrucker.
`-X charset`	gibt den Zeichensatz des Texts an (z. B. `ASCII`, `iso` n, `latin` n).

Zu `a2ps` existiert eine ausführliche `man`-Seite. Einen Überblick über alle Optionen erhalten Sie mit `a2ps -h`. Viele weitere Informationen finden Sie unter *http://www.inf.enst.fr/ demaille/a2ps/*.

enscript `[optionen] quelldatei -p zieldatei`

`enscript` hat eine ähnliche Funktion wie `a2ps`. Es wandelt eine Textdatei in eine PostScript-Datei um. Optional kann der Text auf mehrere Spalten aufgeteilt werden. Bei Programmcode kann optional eine Syntaxhervorhebung durchgeführt werden.

`--columns=`n	n Spalten pro Seite darstellen. Eine zweispaltige Formatierung wird mit `-2` erreicht.
`--color`	Syntaxhervorhebung mit Farben durchführen (muss mit `-E` kombiniert werden)
`-E`	Syntaxhervorhebung durchführen (fett, kursiv)
`-r`	Blatt im Querformat füllen
`-X charset`	Gibt den Zeichensatz des Texts an (z. B. `ascii`, `latin` n)

Viele weitere Optionen sind in der `man`-Seite zu `enscript` beschrieben. Unter *http://people.ssh.fi/mtr/genscript/* finden Sie auch noch weitere Informationen.

6.6.3 Konvertierung HTML → PostScript

Wenn Sie eine HTML-Datei ausdrucken möchten, können Sie dazu einen Webbrowser verwenden. Für die automatische Konvertierung vom HTML- in das PostScript-Format eignet sich das Perl-Script `html2ps` allerdings erheblich besser.

Die Bedienung ist denkbar einfach: `html2ps -D name.html > name.ps`. Die Option `-D` bewirkt, dass `html2ps` DSC-konforme Kommentare in die PostScript-Datei einbaut, was deren Weiterverarbeitung sehr erleichtert. Falls `html2ps` nicht Teil Ihrer Distribution ist, finden Sie den Konverter im Internet unter *http://user.it.uu.se/ jan/html2ps.html*.

6.6.4 Konvertierung PS ↔ PDF

ps2pdf: Das Kommando `ps2pdf quelle.ps ziel.pdf` erzeugt aus einer beliebigen PostScript-Datei eine PDF-Datei. Das Kommando erfüllt damit im Prinzip dieselbe Funktion wie das kommerzielle Programm Adobe Distiller. Es basiert auf GhostScript.

`ps2pdf` erzeugt derzeit Dateien, die zum PDF-Format 1.2 für den Acrobat Reader 3.n kompatibel sind. Die Dokumentation weist aber darauf hin, dass sich das Standardverhalten in Zukunft ändern kann. Wenn Sie die Kompatibilität zu einer bestimmten PDF-Version sicherstellen möchten, sollten Sie die Kommandos `ps2pdf12`, `ps2pdf13` und `ps2pdf14` einsetzen. Die Qualität der PDF-Dateien hängt stark von den im PostScript-Dokument verwendeten Schriftarten ab. Ggf. werden die Zeichen durch Bitmaps ersetzt, was die Darstellungsqualität stark mindert.

`-r`*n* gibt die Auflösung für Bitmap-Fonts an (Einheit *dots per inch*).

`-dPDFSETTINGS=/default` | `/screen` | `/printer` | `/prepress`
: gibt an, ob das PDF-Dokument für eine beliebige Verwendung, für die Bildschirmdarstellung, für einen gewöhnlichen Ausdruck oder für eine Belichtung (Buchdruck etc.) optimiert werden soll. Diese vier Voreinstellungen vermeiden die Einzeleinstellung zahlreicher Optionen.

`-dEncodeColorImages=false`
: verhindert jegliches Komprimieren von Bildern. Die resultierenden PDF-Dateien werden nun sehr groß. Diese Option in Kombination mit der */prepress*-Einstellung liefert aber PDFs in optimaler Druck- bzw. Belichtungsqualität.

Das Verhalten von `ps2pdf` kann durch viele weitere Optionen gesteuert werden. Eine vollständige Dokumentation finden Sie auf der `ps2pdf`-Website *http://stat.tamu.edu/doc/gs/Ps2pdf.htm*. Die genaue Bedeutung der Optionen ist noch ausführlicher in einem technischen Dokument zum Adobe Distiller beschrieben:

http://partners.adobe.com/asn/developer/acrosdk/DOCS/distparm.pdf

pdf2ps: Den umgekehrten Weg geht `pdf2ps quelle.pdf ziel.ps`. Auch dieses Kommando greift auf GhostScript zurück.

6.6.5 PostScript-Tools (psutils)

Mit den meisten PostScript-Viewern können Sie einzelne Seiten markieren und als eigene PostScript-Datei speichern. Wenn Sie solche oder ähnliche Operationen automatisieren möchten, werden Sie sich über die Kommandos des `psutils`-Pakets freuen. Dabei handelt es sich teils um eigenständige Programme, teils um `bash`- oder Perl-Script-Dateien.

In diesem Abschnitt wurde aus Platzgründen auf eine detaillierte Beschreibung der zahlreichen Kommandos verzichtet – aber allein die Auflistung der wichtigsten Kommandos sollte eigentlich ausreichen, um Sie auf den Geschmack zu bringen. Weitere Details finden Sie dann in den `man`-Pages.

psutils-Kommandos	
`epsffit`	passt die Größe einer EPS-Datei an.
`extractres`	analysiert die Datei und liefert `%%IncludeResource`-Kommentare für alle benötigten Fonts, Dateien etc.
`fixfmps`	passt FrameMaker-Dateien an die `psutils`-Konventionen an.
`fixmacps`	passt Macintosh-Dateien an die `psutils`-Konventionen an.
`fixscribeps`	passt Scribe-Dateien an die `psutils`-Konventionen an.
`fixtpps`	passt Troff/Tpscript-Dateien an die `psutils`-Konventionen an.
`fixwfwps`	passt MS Word-Dateien an die `psutils`-Konventionen an.
`fixwpps`	passt WordPerfect-Dateien an die `psutils`-Konventionen an.
`fixwwps`	passt MS Write-Dateien an die `psutils`-Konventionen an.
`getafm`	erzeugt AFM-Dateien zur Beschreibung von Fonts.
`includeres`	fügt die mit `extractres` erzeugten Kommentare in eine PostScript-Datei ein.
`psbook`	ordnet die Seiten eines Textes so an, dass ganze Bögen (etwa mit je 16 Seiten) gedruckt werden können.
`psnup`	ordnet mehrere verkleinerte Seiten auf einem Blatt an.
`psresize`	verändert die erforderliche Papiergröße eines Dokuments; das Kommando löst das regelmäßig auftretende Problem des Ausdrucks von PostScript-Dokumenten, die für das US-Letter-Format erzeugt wurden.
`psselect`	extrahiert einzelne Seiten aus einer PostScript-Datei.
`pstops`	ordnet die Seiten eines Dokuments in einer neuen Reihenfolge.

Das folgende Beispiel zeigt, wie eine mit LaTeX und DVIPS erzeugte PostScript-Datei mit dem Manuskript dieses Buches in eine Darstellung mit 64 Seiten pro Blatt umgewandelt wird. Damit erscheint jede Seite nur noch briefmarkengroß, was anschließend in einem PostScript-Viewer eine rasche, übersichtsartige Kon-

trolle des Seitenlayouts ermöglicht (ähnlich wie die Druckvorschau bei Microsoft Word mit dem kleinstmöglichen Zoomfaktor):

```
user$  psnup -b-0.4cm -64 -q < linux.ps > vorschau.ps
```

Die obigen Kommandos funktionieren leider nur dann, wenn die PostScript-Dateien DSC-konforme Kommentare enthalten. (DSC steht für *Document Structuring Conventions*. Die Kommentare werden nicht ausgedruckt, enthalten aber wichtige Informationen über die Größe einer Seite, über den Beginn und das Ende von Seiten etc.) EPS-Dateien sind einseitige PostScript-Dateien, die spezielle Kommentare zur Einbettung in andere Dokumente enthalten (insbesondere Bounding-Box-Angaben über die Größe des Ausdrucks).

Die Bearbeitung von PostScript-Dateien, die unter Windows erzeugt wurden, ist generell eine mühsame Angelegenheit; auch die aufgezählten Kommandos führen nicht immer zu den erwünschten Ergebnissen. Eine Chance auf Erfolg haben Sie in jedem Fall nur dann, wenn Sie beim Windows-Druckertreiber die Option zur Erzeugung von DSC-konformen PostScript-Dateien aktivieren.

6.6.6 GhostScript, Gimp-Print und andere Druckertreiber

GhostScript (`gs`) ist in der Lage, PostScript- und PDF-Dokumente in zahlreiche andere Formate zu konvertieren. `gs` greift dabei auf eine Sammlung von Schriftarten zurück (`/usr/share/ghostscript/fonts`), um PostScript-Fonts in eine Bitmap-Darstellung umzuwandeln. Es ist ein wichtiger Baustein eines jeden Drucker-Systems, weil es den Ausdruck von PostScript-Dokumenten auf gewöhnlichen Druckern ermöglicht. Das Programm wird aber auch von vielen in diesem Kapitel vorgestellten PostScript-Viewern und -Konvertern eingesetzt. Es gibt drei GhostScript-Versionen mit unterschiedlichen Lizenzen:

- **AFPL GhostScript:** Die jeweils neueste Version heißt „AFPL GhostScript *n*“. Die Version unterliegt der *Aladdin Free Public License* (AFPL). Das bedeutet, dass diese Version zwar über das Internet frei verfügbar ist, eine kommerzielle Verbreitung aber nur nach einer Lizenzvereinbarung möglich ist.
- **GNU GhostScript:** Ältere GhostScript-Versionen bekommen nach einer Weile den Namen „GNU GhostScript *n*“. Für sie gilt dann die GPL, d. h., die Verbreitung unterliegt (fast) keinen Einschränkungen. Die meisten Distributionen liefern daher diese Version aus. Für den Anwender hat das den Nachteil, dass die jeweils neuesten Treiber noch nicht enthalten sind.

  Eine Variante von GNU GhostScript ist **ESP GhostScript**, wobei ESP für den Firmennamen *Easy Software Products* steht. Diese Version ist speziell für die Zusammenarbeit mit CUPS optimiert und enthält zusätzliche Patches und Updates (*http://www.cups.org/GhostScript.php*).

- **Artifex GhostScript:** Hierbei handelt es sich um eine kommerziell lizenzierte OEM-Variante von AFPL GhostScript, die beispielsweise an Druckerhersteller verkauft wird.

Mehr Informationen zu GhostScript finden Sie auf den Webseiten *http://www.cs.wisc.edu/ ghost/*, *http://www.GhostScript.com/* und *http://www.artifex.com/*.

Bedienung: Im Regelfall wird `gs` automatisch vom Spooling-System aufgerufen, um die PostScript-Daten in das Format des jeweiligen Druckers umzuwandeln. Ebenso greifen alle PostScript- und PDF-Viewer (außer Acrobat Reader) und viele Grafikkonverter auf `gs` zurück. Auf der Kommandozeile wirde es mit `gs [optionen] postscriptdatei [quit.ps]` aufgerufen. Damit `gs` korrekt funktioniert, müssen mindestens zwei Optionen angegeben werden: `-sOutputFile=` zur Angabe der Datei, in die das Ergebnis geschrieben werden soll, sowie `-sDEVICE=`*name* oder `@`*name.upp* zur Einstellung des Ausgabeformats. In der Regel ist es sinnvoll, auch die Option `-dNOPAUSE` zu verwenden. Falls Sie auf DIN-A4-Papier drucken möchten, sollten Sie schließlich noch `-sPAPERSIZE=a4` angeben.

Wie bereits erwähnt, gibt es Druckertreiber, die Teil des GhostScript-Projekts sind, und solche, die aus anderen Projekten zu GhostScript hinzugefügt wurden. Daraus ergibt sich, dass es mehrere unterschiedliche Methoden zur Angabe des Druckertyps gibt. Wesentlich unangenehmer ist aber, dass jeder der externen Treiber auch eigene Optionen zur Einstellung von Zusatzparametern (Auflösung, Seitengröße etc.) aufweist. Eine Beschreibung aller `gs`-Optionen ist aus Platzgründen unmöglich. Die treiberspezifischen Optionen sind in der jeweiligen Dokumentation beschrieben.

- `-sDEVICE=`*name*
  Das ist die klassische Methode zur Angabe des Druckermodells. Wichtige Treibernamen sind z. B. `eps9high`, `ljet4` oder `djet820c`. PostScript-Dateien, die nur eine einzige Seite (ein Bild) enthalten, können auch in diverse Grafikformate konvertiert werden. Eine Liste aller Device-Treiber liefert `gs -h`. Mit der Zusatzoption `-r` kann die gewünschte Auflösung in *dots per inch* angegeben werden.

- `@`*name.upp*
  Für viele Tintenstrahldrucker gibt es so genannte Uniprint-Treiber, die ebenfalls Teil des GhostScript-Projekts sind. Mit `@` wird der Name der `*.upp`-Datei angegeben.

- `-sDEVICE=stp -sModel=`*name*
  Zahlreiche Drucker (überwiegend Tintenstrahldrucker) können auch mit den Druckertreibern des Gimp-Print-Projekts angesteuert werden.

- `-sDEVICE=hpijs -sDeviceName=`*name*
  Speziell für HP-Drucker gibt es eigene HP-Treiber. Als Device-Name sind

z. B. die folgenden Zeichenketten erlaubt: `DJ630`, `DJ6xx`, `DJ6xxPhoto`, `DJ8xx`, `DJ9xx` oder `DJ9xxVIP`.

- `-sDEVICE=hplj -sModel=`*name*
  Das ist ein weiterer Treiber für alle HP-Drucker, die intern PCL3+ als Druckersprache verwenden (z. B. HP DeskJet 500, 500C, 510, 520, 540, 550C, 560C, 850C und 855C).

- `-sDEVICE=pdfwrite -dCompatibilityLevel=n`
  GhostScript kann auch PDF-Dateien erzeugen. *n* gibt an, zu welcher PDF-Version das Ergebnis kompatibel sein soll. Momentan werden die Werte 1.2 (für Acrobat Reader 3), 1.3 (Acrobat 4) und 1.4 (Acrobat 5) unterstützt.

- `-sDEVICE=pswrite -dLanguageLevel=n`
  Auf den ersten Blick wirkt es absurd, dass GhostScript auch PostScript-Dokumente erzeugen kann. Eigentlich ist es ja die Aufgabe von GhostScript, solche Dokumente zu verarbeiten. Diese Variante von GhostScript ist aber ausgesprochen nützlich, wenn Sie ein Dokument gemäß PostScript Level 3 auf einem alten Seitendrucker ausgeben möchten, der nur PostScript Level 1 versteht. *n* gibt den gewünschten PostScript-Level für die Ausgabe an.

Welche Druckermodelle `gs` unterstützt, hängt davon ab, mit welchen Zusatzfiltern `gs` bei Ihrer Distribution ausgeliefert wurde. Die weiteren Optionen sind vom Druckertreiber unabhängig:

- `-dBATCH`
  `gs` arbeitet normalerweise interaktiv und meldet sich nach der Bearbeitung der letzten Seite mit einer Eingabe-Aufforderung. Um `gs` klarzumachen, dass die Übersetzung beendet ist, müssen Sie (Strg)+(D) eingeben. Die Option `-dBATCH` vermeidet diese Eingabe-Aufforderung.

- `-dNODISPLAY`
  verzichtet auf die Anzeige von diversen Meldungen auf dem Bildschirm. Außerdem ist es nun nicht mehr notwendig, die Vollendung einer jeden Seite mit (↵) zu bestätigen.

- `-sPAPERSIZE=`*Papierformat*
  bestimmt das Papierformat der Ausgabe (beispielsweise `a4` für das DIN-A4-Format). Per Default verwendet `gs` das amerikanische Letter-Format, das etwas kleiner ist. Beachten Sie, dass manche Druckertreiber eigene Optionen zur Einstellung der Papiergröße haben.

- `-dSAFER`
  Diese Option schließt ein Sicherheitsloch. Sie muss unbedingt verwendet werden, wenn `gs` vom Spooling-System als Output-Filter verwendet wird.

- `-q`
  unterdrückt diverse Statusmeldungen (*quiet).*

- `-sOutputFile=`*Dateiname*
  schreibt das Ergebnis der Konvertierung in die angegebene Datei.

Die folgende Anweisung übersetzt `test.ps` in das Format des HP-Laserjet 3. Das Ergebnis wird in die Datei `test.hp` geschrieben:

```
user$ gs -sDEVICE=ljet3 -sOutputFile=test.hp \
  >    -sPAPERSIZE=a4 -dNOPAUSE -dBATCH test.ps
```

Das zweite Beispiel wandelt eine PostScript- in eine PDF-Datei um. (Noch bequemer können Sie das mit dem Kommando `ps2pdf` erledigen. Dabei handelt es sich um ein kleines Script, das auf `gs` zurückgreift.)

```
user$ gs -dNOPAUSE -dBATCH -sDEVICE=pdfwrite \
  >    -sOutputFile=test.pdf input.ps
```

Externe GhostScript-Druckertreiber: Eine Menge von Konvertern bzw. Druckertreibern, die via GhostScript eingesetzt werden, wurden eigentlich außerhalb des GhostScript-Projekts entwickelt. Zum Teil überlappen sich die GhostScript-Orginaltreiber mit den hier beschriebenen Projekten, weswegen es mittlerweile für viele Drucker mehrere Treiber gibt. Allerdings hängt es von der Linux-Distribution ab, welche Treiber tatsächlich mitgeliefert werden. Im Folgenden werden nur die wichtigsten derartigen Projekte aufgezählt.

- **Gimp-Print alias stp:**
  Einige Gimp-Anwender mit modernen Tintenstrahldruckern, die mit den GhostScript-Druckertreibern unzufrieden waren, begannen vor Jahren mit der Entwicklung eigener Druckertreiber. Diese waren anfänglich nur für Gimp gedacht und nur für Epson-Stylus-Modelle konzipiert. Im Laufe der Zeit wurden Treiber für zahlreiche weitere Tintenstrahl- und Laserdrucker von Canon, HP etc. integriert. Außerdem wurden die Treiber so angepasst, dass sie auch via GhostScript bzw. direkt von CUPS genutzt werden konnten. Die Gimp-Print-Druckertreiber stellen die qualitativ beste Möglichkeit für den Fotodruck unter Linux dar (Infos: `man gs-stp` und *http://gimp-print.sourceforge.net/*).
- **Hewlett Packard Inkjet Server alias hpijs:**
  HP hat selbst Druckertreiber für die meisten seiner Tintenstrahlmodelle entwickelt (Info: *http://hp.sourceforge.net* und *http://hpinkjet.sourceforge.net/*).
- **HPDJ-Treiber (PCL3-Treiber):**
  Lange bevor HP eigene Treiber für Linux zur Verfügung stellte, wurden viele HP-Modelle durch den HPDJ-Treiber unterstützt. Dieser Treiber wurde später in PCL3 umbenannt, weil er alle Drucker unterstützt, die intern die Druckersprache PCL3+ verwenden. Der Treiber ist gut dokumentiert (Info: `man gs-hpdj` und *http://home.t-online.de/home/Martin.Lottermoser/pcl3.html*).

- **IBM Omni:** IBM hat den Code seiner ursprünglich für OS/2 entwickelten Druckertreiber unter die Open-Source-Lizenz (LGPL) gestellt. Die Integration in `gs` befindet sich noch im Beta-Stadium, weswegen die Druckertreiber momentan nicht mit allen Distributionen mitgeliefert werden (Info: *http://sourceforge.net/projects/omniprint*).

Kapitel 7

Audio und Video

Einführung

Dieses Kapitel beschäftigt sich damit, wie Sie unter Linux Audio-CDs, Audio-Dateien (MP3, Ogg Vorbis, RAM) und Video-DVDs abspielen können. Im Detail werden die folgenden Themen behandelt:

Konfiguration einer Soundkarte
Kernel-Soundsysteme (OSS, ALSA)
Sound-Server (KDE, Gnome)
Abspielen von Audio-CDs
Audio-Formate (MP3, Ogg Vorbis, RealAudio)
Audio-Tracks einlesen (rippen)
Video-DVDs abspielen
Einlesen und Scannen von Bilddateien
Bilddateien anzeigen
CDs und DVDs brennen

Weitergehende Multimedia-Funktionen – etwa Mehrkanal-Audio (Surround-Klang), MIDI-Anwendungen, TV-Tuner-Karten, die Verwendung von Linux als digitaler Video-Rekorder etc. – werden in diesem Buch allerdings nicht behandelt. Einen guten Startpunkt für die weitere Recherche zum Thema Linux und Multimedia bieten die Websites
http://m4l.berlios.de/,
http://www.tldp.org/HOWTO/Sound-HOWTO/,
http://www.linuxdj.com/audio/quality/,
http://sound.condorow.net/ und
http://www.exploits.org/v4l/.
Außerdem gibt es mittlerweile eine ganze Reihe von Büchern, die sich explizit mit dem Thema Linux und Multimedia beschäftigen.

7.1 Konfiguration der Soundkarte

Die Konfiguration von Sound-Karten war unter Linux lange relativ kompliziert. Mittlerweile hat sich das zum Glück geändert. Die meisten marktüblichen Sound-Karten werden bereits während der Installation von Linux richtig konfiguriert. Nur wenn das nicht klappt, müssen Sie eventuell eines der folgenden Konfigurationswerkzeuge einsetzen; bei RedHat und Fedora das Programm `system-config-soundcard`, bei Suse das YaST-Modul *Hardware* → *Sound* oder für die ALSA-Konfiguration `alsaconf`. Manche On-Board-Soundsysteme verweigern sich auch dann noch. In einem solchem Fall sollte man erwägen, dem Computer eine Standard-Soundkarte zu spendieren.

Hinter den Kulissen sieht die Ansteuerung der Soundkarte folgendermaßen aus: Auf unterster Ebene befinden sich ein oder mehrere Kernel-Module, die die Soundkarte direkt ansprechen. Dabei stehen zwei Typen von Kernel-Modulen zur Auswahl: OSS-Module und ALSA-Module (siehe Seite 287) – je nach Distribution. In keinem Fall müssen die Audio-Programme die Soundkarte direkt ansprechen, sondern sie können dazu ein Audio-Device verwenden (meistens `/dev/dsp`). Grundsätzlich kann immer nur ein Audio-Programm das Device nutzen. Ein weiteres Audio-Programm kann erst dann Daten an das Device senden, wenn das erste Programm fertig ist.

Wenn Sie unter KDE oder Gnome arbeiten, wird es etwas komplizierter: Hier steht ein so genannter Sound-Server zwischen den Kernel-Modulen und den Audio-Programmen. Er ermöglicht, ähnlich wie der Printer-Spooler, mehreren Programmen den *gleichzeitigen* Zugang zum Audio-System (siehe Seite 290). Generell müssen Sie nach der erstmaligen Konfiguration der Soundkarte KDE bzw. Gnome neu starten, damit die Änderungen auch im Desktop aktiv werden.

Tipp

Bei manchen Audio-Programmen (z. B. `xmms`, `freeamp` etc.) können Sie im Konfigurationsdialog einstellen, auf welche Weise die Soundkarte angesprochen werden soll. Wenn ein Programm auf Anhieb nicht läuft, liegt es oft an Inkompatibilitäten zwischen dem Programm und dem auf Ihrem System aktiven Soundsystem. Die richtige Konfiguration behebt solche Probleme.

7.1.1 Lautstärkeregelung

Bevor ein Ton aus dem Lautsprecher kommt, muss die Lautstärke richtig eingestellt werden. Dazu kommen je nach Soundsystem unterschiedliche Programme zum Einsatz (Natürlich muss der Regler an der Lautsprecherbox selbst auch etwas offen sein).

KDE: `kamix`
Gnome: `gnome-volume-control`

OSS: `aumix`, `xaumix`
ALSA: `gamix`, `alsamixer` (Bedienung mit Cursortasten)

Aufgrund diverser Kompatibilitätsfunktionen in den einzelnen Soundsystemen funktioniert in den meisten Fällen beinahe jeder Mixer. Ein gemeinsames Merkmal all dieser Programme ist, dass eine ganze Menge von Audio-Kanälen zusammengemischt werden können. Für normale Anwendungen sind aber nur drei Kanäle wichtig:

- **Master-Lautstärke** (Master volume): Dieser Regler (üblicherweise der erste) steuert die Lautstärke des Gesamtsignals.
- **PCM-Lautstärke:** PCM steht für *Pulse Code Modulation*. Der PCM-Kanal wird von allen Audio-Programmen verwendet, die selbst Audio-Daten in diesem Format erzeugen (z. B. allen MP3-Playern). Wenn Sie eine beliebige Audio-Datei abspielen, darf weder die Master- noch die PCM-Lautstärke auf 0 stehen.
- **CD-Lautstärke:** Wenn Sie Audio-CDs abspielen möchten, steuert dieser Kanal, wie stark die direkt vom CD-Laufwerk kommenden Daten in das Gesamtsignal einfließen sollen. Damit Sie also CDs anhören können, müssen sowohl die Master- als auch die CD-Lautstärke eingestellt werden.

Welche weiteren Kanäle es gibt (Mikrofon, Video etc.) und wie sie zu steuern sind, hängt von den Mixer-Fähigkeiten Ihrer Soundkarte und vom Soundsystem ab.

7.1.2 Erster Test

Als ersten Test der Sound-Kartenkonfiguration spielen Sie einfach irgend eine `*.wav`-Sounddatei ab. (Finden kann man Dateien mit `locate` oder `find`.) Welches Kommando Sie zum Abspielen der Sounddatei verwenden müssen, hängt davon ab, welchen Desktop Sie verwenden (wenn Sie keinen Desktop verwenden, hängt es davon ab, welches Soundsystem Sie einsetzen). `artsplay` (KDE), `esdplay` (Gnome), `play` (OSS = *Open Sound System*) oder `aplay` (ALSA = *Advanced Linux Sound Architecture*).

7.1.3 Sound-Kernel-Module

Die Ansteuerung der Soundkarte unter Linux erfolgt durch Kernel-Module. Aber wie so oft unter Linux gibt es hierfür nicht eine Lösung, sondern gleich mehrere (Wir erwähnen hier auch noch ältere Systeme wie OSS und die Kernel-Version 2.4, da Debian diese Version immer noch mitbringt):

- **OSS/Free alias OSS/Lite:** OSS/Free (*OSS = Open Sound System*) ist bis Version 2.4 in den Standard-Kernelcode integriert (*http://www.linux.org.uk/OSS/*).
- **OSS/Linux:** Das ist die kommerzielle Variante zu OSS/Free. Der wesentliche Unterschied besteht darin, dass es Support gibt und einige zusätzliche Sound-Karten bzw. besondere Eigenschaften von Sound-Karten unterstützt werden (*http://www.opensound.com/*).
- **ALSA:** ALSA ist der Nachfolger von OSS/Free. ALSA ist in den Kernel 2.6 integriert. Einige Distributionen (Suse) setzen ALSA schon seit Jahren ein, also auch unter Kernel 2.4 (*http://www.alsa-project.org/*).
- **Herstellerspezifische Treiber:** Neben diesen Komplettpaketen, in denen versucht wird, möglichst viele Sound-Karten abzudecken, gibt es vereinzelt auch herstellerspezifische Treiber. Das Problem dieser Treiber besteht darin, dass sie oft nur in binärer Form vorliegen und damit an eine ganz bestimmte Kernel-Version gebunden sind.

Allen Systemen ist gemeinsam, dass die Funktionen in Form von Kernel-Modulen realisiert sind. Linux kann Ihre Soundkarte nur ansprechen, wenn die entsprechenden Module geladen wurden. Damit das automatisch klappt, muss entweder `/etc/modules.conf` für Kernel 2.4 bzw. `/etc/modprobe.conf` für Kernel 2.6 richtig eingestellt werden oder die Module werden im Rahmen des Init-V-Prozesses geladen. Die beiden folgenden Abschnitte geben einige Tipps zur Konfiguration durch die Dateien `modules.conf` bzw. `modprobe.conf`. Diese Informationen sind dann hilfreich, wenn die Sound-Kartenkonfiguration mit den Programmen Ihrer Distribution nicht gelingt. Inzwischen wird OSS zunehmend von ALSA verdrängt, weshalb wir uns hier auch auf ALSA beschränken.

7.1.4 ALSA-Konfiguration

ALSA basiert wie OSS auf Kernel-Modulen. Der Zugriff auf diverse Soundfunktionen erfolgt über Dateien im Verzeichnis `/proc/asound`. Wenn das Verzeichnis `/proc/asound` existiert, wissen Sie also, dass Ihre Distribution ALSA unterstützt.

Zur Konfiguration von ALSA verwenden Sie entweder das von Ihrer Distribution mitgelieferte Konfigurationsprogramm oder das zu ALSA gehörende Programm `alsaconf`. Anschließend können Sie die Lautstärke mit `alsamixer` oder `amixer` einstellen und eine WAV-Datei mit `aplay` abspielen. `alsactl` ermöglicht eine weitergehende Steuerung diverser Spezialfunktionen von Sound-Karten.

Normalerweise kümmert sich ein Init-V-Script darum, dass die ALSA-Module beim Systemstart automatisch geladen werden (z.B. `/etc/init.d/alsasound`). Bei Problemen kann das Soundsystem damit auch neu gestartet werden. Falls Sie unter KDE oder Gnome arbeiten, sollten Sie sich anschließend neu einloggen.

Kernel 2.4: Die folgenden Zeilen zeigen als Beispiel die Einstellungen in `/etc/modules.conf` (beispielhaft für den Sound-Chip eines Mainboards der Firma VIA). Nur die ersten vier Zeilen betreffen die eigentliche ALSA-Konfiguration. Die restlichen Einstellungen sorgen dafür, dass Programme, die OSS-Funktionen nutzen, auch weiterhin funktionieren.

```
# in /etc/modules.conf
# ALSA-Konfiguration
options snd-via82xx snd_ac97_clock=48000 snd_enable=1 snd_index=0
options snd snd_cards_limit=1 snd_major=116
alias   char-major-116 snd
alias   snd-card-0 snd-via82xx

# OSS-Kompatibilität
alias sound-slot-0       snd-card-0
alias sound-service-0-0  snd-mixer-oss
alias sound-slot-1       off
alias sound-service-1-0  off
alias sound-slot-2       off
alias sound-service-2-0  off
alias sound-slot-3       off
alias sound-service-3-0  off

alias sound-service-0-0  snd-mixer-oss
alias sound-service-0-1  snd-seq-oss
alias sound-service-0-3  snd-pcm-oss
alias sound-service-0-8  snd-seq-oss
alias sound-service-0-11 snd-mixer-oss
alias sound-service-0-12 snd-pcm-oss
```

Kernel 2.6: Bei Kernel 2.6 haben sich die Namen mancher Optionen geändert. `snd_` muss ersatzlos entfernt werden. Ansonsten sieht die Konfiguration in `modprobe.conf` ganz ähnlich aus:

```
# in /etc/modprobe.conf
# ALSA-Konfiguration
options snd-via82xx ac97_clock=48000 enable=1 index=0
options snd cards_limit=1 major=116
alias char-major-116 snd
alias snd-card-0 snd-via82xx

# OSS-Kompatibilität
alias sound-slot-0       snd-card-0
alias sound-service-0-0  snd-mixer-oss
alias sound-service-0-1  snd-seq-oss
alias sound-service-0-3  snd-pcm-oss
alias sound-service-0-8  snd-seq-oss
alias sound-service-0-11 snd-mixer-oss
alias sound-service-0-12 snd-pcm-oss
```

7.1.5 Die GStreamer-Bibliothek

GStreamer ist ein Open Source Multimedia Framework. Diese Bibliothek stellt grundlegende Funktionen zur Verarbeitung von Datenströmen aller Art bereit und kann so als Basis von Multimedia-Anwendungen wie z.B. Media-Player oder Audio-Mixer dienen. Sie besteht aus einer Reihe von Codecs und Plugins zum Einlesen, Ausgeben, Dekodieren, Enkodieren usw. Sie kann einfach mithilfe von Plugins erweitert und so den Bedürfnissen der jeweiligen Anwendung angepasst werden. Mittlerweile basieren zahlreiche Programme auf GStreamer, darunter sind die Gnome-Desktop-Umgebung sowie dazu passende Musik- und Videoprogramme.

Der Vorteil von GStreamer liegt in der einfachen, sauberen API, die den Entwicklern ein möglichst leicht bedienbares Grundgerüst bereitstellt, so dass weniger Zeit in die Entwicklung von grundlegenden Funktionen investiert werden muss. Damit gewinnt der Entwickler Zeit für die Programmteile, welche die neue Funktionalität des Programms ausmachen. Weitere Codecs und Plugins lassen sich leicht zu GStreamer hinzufügen.

7.1.6 KDE- und Gnome-Soundsystem

Wenn Sie unter KDE oder Gnome arbeiten, steht zwischen den Kernel-Soundmodulen und den einzelnen Anwendungen oft eine zusätzliche Bibliothek, die unter anderem die folgenden Aufgaben erfüllen soll:

- Alle Programme sollen gleichzeitig Zugriff auf das Audio-System bekommen, ohne sich gegenseitig zu blockieren. Dazu müssen die Audio-Signale aller Programme zu einem einzigen Signal vereint (überlagert) werden. Beispielsweise ist es praktisch, wenn beim Eingang einer neuen E-Mail auch dann eine Melodie abgespielt werden kann, wenn Sie gerade eine MP3-Datei anhören.
- Das Soundsystem soll netzwerktauglich sein. Wenn das funktioniert, können Sie sich auf einem Rechner im lokalen Netzwerk einloggen und dort ein Audio-Programm starten. Tatsächlich hören Sie die Klänge aber auf Ihrem lokalen Rechner.
- Das resultierende Gesamtsignal soll aufgezeichnet werden können (recording-Funktion).

Bei KDE übernimmt das Soundsystem „aRts“ diese Aufgaben. Das äquivalente Gnome-System hat den Namen „EsounD“. Allerdings sind diese Programme nicht ganz unproblematisch: Was geschieht, wenn ein Programm, das weder der KDE- noch der Gnome-Welt angehört, direkt das Soundsystem des Kernels anspricht? Was passiert, wenn ein Gnome-Audio-Programm unter KDE ausgeführt werden soll (oder umgekehrt)?

Die vielen Kombinationsmöglichkeiten aus zwei Kernelsy-Stemen (OSS, ALSA), zusätzlichen herstellerspezifischen Treibern und zwei verschiedenen Sound-Servern (KDE, Gnome) führen leider manchmal dazu, dass der Lautsprecher trotz richtig konfigurierter Sound-Karte still bleibt bzw. dass die Audio-Qualität unter Ihren Erwartungen liegt. In diesem Fall kann es sinnvoll sein, den Sound-Server zu deaktivieren. Zwar ist es nun meist unmöglich, dass mehrere Audio-Programme gleichzeitig laufen, aber dafür erzielen Sie vielleicht gerade mit Ihrem Lieblings-MP3-Player einen besseren Klang.

Sound-Server deaktivieren: Sowohl bei KDE als auch bei Gnome können Sie den automatischen Start des Sound-Servers unterbinden. Wie weit KDE- bzw. Gnome-Programme nun noch in der Lage sind, die Soundkarte anzusprechen, hängt vom jeweiligen Programm ab.

aRts (KDE-Soundsystem): aRts steht für *analog real-time synthesizer*. aRts besteht aus einer Reihe von Modulen, mit denen Audio-Daten erzeugt, gefiltert, vereint etc. werden. KDE-Programme sprechen aRts über den Dämon `artsd` an, der zusammen mit KDE gestartet wird. Programme, die nicht aRts-kompatibel sind, sondern direkt auf die Devices der Kernel-Soundmodule zugreifen, werden automatisch via `artsdsp` umgeleitet. Damit sollten sich auch diese Programme nahtlos in das KDE-System einfügen. Ausführliche Informationen zu aRts finden Sie unter *http://www.arts-project.org/*.

aRts ist per Default aktiviert und wird daher zusammen mit KDE gestartet. Wenn Sie das nicht möchten oder wenn Sie einzelne `artsd`-Parameter konfigurieren möchten (das ist selten erforderlich), starten Sie das KDE-Kontrollzentrum und öffnen den Dialog *Sound und Multimedia* → *Soundsystem*.

Ob und mit welchen Melodien diverse KDE-Aktionen begleitet werden sollen, können Sie im Dialog *Sound und Multimedia* → *Systemnachrichten* einstellen. Zur Lautstärkeregulierung verwenden Sie das bereits erwähnte Programm `kamix` (KDE-Menükommando *Multimedia* → *Lautstärkeregler*).

EsounD (Gnome-Soundsystem): EsounD steht für *Enlightened Sound Daemon*. EsounD-kompatible Programme senden Audio-Daten an den Dämon `esd`, der zusammen mit Gnome gestartet wird. Programme, die nicht EsounD-kompatibel sind, sondern direkt auf die Devices der Kernel-Soundmodule zugreifen, werden automatisch via `edsdsp` umgeleitet. Weitere Informationen zu EsounD finden Sie auf der folgenden, leider vollkommen veralteten Seite *http://www.tux.org/ ricdude/EsounD.html*.

Im Programm `gnome-sound-properties` (Gnome-Menü *Präferenzen* → *Audio*) können Sie einstellen, ob `esd` zusammen mit Gnome gestartet werden soll (diese Einstellung gilt erst beim nächsten Gnome-Start!) und ob und wie Gnome-Ereignisse (z. B. Login, Logout etc.) von Geräuschen begleitet werden sollen.

7.2 CD-Player

Die Aufgabe eines CD-Players ist ziemlich trivial: Das Programm soll das Inhaltsverzeichnis einer Audio-CD einlesen und die Audio-Tracks dann abspielen. Das Programm überlässt dabei dem CD-Laufwerk praktisch die gesamte Arbeit (also das Auslesen der Daten, die Fehlerkorrektur, die Umwandlung in analoge Signale). Der CD-Player erfüllt also nur Steuerungsaufgaben (und führt nicht wie bei einem MP3-Player auch eine Dekodierung durch). Wie bei einem „richtigen“ CD-Player bieten auch deren virtuelle Gegenstücke Buttons zur Navigation innerhalb eines Tracks bzw. zum Vor- und Zurückspringen.

Damit Sie die abgespielte Musik tatsächlich hören, muss das CD-Laufwerk über ein Kabel mit der Soundkarte bzw. mit dem Mainboard Ihres Rechners verbunden sein. Außerdem muss die Soundkarte korrekt konfiguriert sein. Weder die Lautstärke des Gesamtsignals (Master) noch die des CD-Kanals dürfen auf 0 stehen. Außerdem muss die Soundkarte mit Boxen, Kopfhörern oder mit Ihrer Stereo-Anlage verbunden sein.

Im Gegensatz zu Daten-CDs werden Audio-CDs nicht in das Dateisystem eingebunden. Der Zugriff auf die CD erfolgt normalerweise direkt über das Device. Die meisten CD-Player gehen davon aus, dass `/dev/cdrom` auf das richtige Device zeigt, beispielsweise auf `/dev/hdc`.

Die Bedienung der diversen Player ist so einfach, dass hier keine weitere Beschreibung erforderlich ist. Die folgende Liste nennt einige Player (ohne Anspruch auf Vollständigkeit):

`cdplayer`, `tcd` und `workbone` können im Textmodus verwendet werden, `xcdplayer` hat eine einfache X-Oberfläche, `xsadp` ist ein Player für X mit Graphics Equalizer, `kscd` stellt die KDE-Variante dar und `gnome-cd` läuft unter Gnome. `grip` ist ein weiterer Gnome-Player, der außerdem Audio-Tracks digital einlesen und diese in MP3-Dateien umwandeln kann.

CDDB und freedb

Eine Besonderheit vieler CD-Player besteht darin, dass sie (sofern eine Internetverbindung besteht), Kontakt mit einem CDDB-Server aufnehmen (*CD database*), in der Regel mit *http://www.freedb.org*. Auf diesem Server befindet sich eine Datenbank, die zu allen dort registrierten CDs den Titel, die Gruppe bzw. die Interpreten sowie das Inhaltsverzeichnis in Textform speichert. Wenn Ihre CD bei *freedb.org* bekannt ist, zeigt der CD-Player also nicht mehr einfach die Track-Nummer an, sondern Titel und Interpret. Die Erkennung der CD basiert auf einem ID-Wert, der sich aus den Längen der Tracks der CD errechnet. Damit CDDB funktioniert, müssen Sie im Konfigurationsdialog des CD-Players CDDB aktivieren (wenn dies nicht schon per Default der Fall ist) und die Adresse für die Datenbankabfragen eintragen: `freedb.freedb.org` mit IP-Port: `8880` (cddbp).

`freedb.freedb.org` leitet die Anfragen automatisch an einen von mehreren Mirror-Servern weiter. Sie können aber statt `freedb.freedb.org` selbst die Adresse eines in Ihrer Nähe befindlichen freedb-Servers angeben, wodurch die Anfragen unter Umständen ein wenig schneller beantwortet werden. Eine Liste aller freedb-Server finden Sie bei *www.freedb.org.* Ist der IP-Port 8880 durch eine Firewall blockiert, können Sie *freedb.org* auch über HTTP ansprechen. In diesem Fall sind folgende Angaben erforderlich: Adresse: `freedb.freedb.org`, IP-Port: 80 und CGI-Script: ~cddb/cddb.cgi.

7.3 Audio-Tools

Da unkomprimierte Audio-Dateien (z. B. im Format WAV oder AU) zur Weitergabe zu groß sind, wurden in der Vergangenheit mehrere komprimierte Audio-Formate entwickelt, darunter MP3, Ogg Vorbis und RealAudio. Das gemeinsame Ziel dieser Formate besteht darin, Audio-Dateien so stark wie möglich zu komprimieren, ohne dabei hörbare Qualitätsverluste in Kauf zu nehmen.

Neben den Formaten WAV, AU, MP3, Ogg Vorbis und RealAudio gibt es diverse andere, die unter Linux zurzeit nicht direkt unterstützt werden. Dazu zählt unter anderem auch das Windows-Media-Audio-Format (WMA). Sie können derartige Dateien aber auch unter Linux anhören, wenn Sie zum Abspielen nicht einen Audio-Player verwenden, sondern die primär für Video-Anwendungen gedachten Programme `xine` oder `mplayer`. Diese Programme sind in der Lage, für Windows gedachte Codecs auch unter Linux zu nutzen.

Bevor wir uns den Tools zuwenden, ein paar Informationen zu wichtigen Begriffen.

Encoder wandeln unkomprimierte Audio-Dateien in ein komprimiertes Format um. Die Kompression soll natürlich möglichst kompakte Ergebnisse bei geringen Qualitätseinbußen liefern – ein sehr rechenintensiver und verhältnismäßig langsamer Vorgang.

Decoder sorgen für die Umwandlung der komprimierten Daten in ein Format, das an Sound-Karten weitergegeben werden kann. Jeder Audio-Player muss daher auf Decoder für das jeweilige Format zurückgreifen. Decoder benötigen Sie aber auch, wenn Sie komprimierte Dateien in ein unkomprimiertes Format zurückverwandeln möchten.

Streaming: kommt zum Einsatz, wenn Audio-Dateien direkt beim Surfen angehört werden sollen. Die Wiedergabe der Musik soll dabei sofort nach dem Start der Übertragung beginnen und nicht erst nach dem Ende der Übertragung. Außerdem werden die Daten meist deutlich höher komprimiert, was sich zwar negativ auf die Tonqualität auswirkt, dafür aber selbst bei einer Internet-Verbindung via Modem oder ISDN-Karte eine Direktübertragung ermöglicht. Viele in diesem Buch vorgestellte Player kommen nur mit MP3- oder Ogg-Vorbis-Streams

zurecht. Daneben gibt es aber eine Menge weiterer Streaming-Formate. Derartige Dateien können Sie nur mit dem kostenlosen RealPlayer bzw. mit den Programmen `xine` bzw. `mplayer` anhören.

ID3-Tags: Am Ende von MP3-Dateien sind üblicherweise Informationen über den Titel gespeichert (Name des Titels, Gruppe oder Interpreten etc.) Beim MP3-Format werden diese Informationen als ID3-Tags bezeichnet. Andere Audio-Formate bieten ebenfalls die Möglichkeit, entsprechende Informationen zu speichern. Viele Audio-Player lesen diese Information aus und zeigen sie während der Wiedergabe an.

M3U-Format: Ein gemeinsames Merkmal der meisten Audio-Player besteht darin, dass sie Listen von Audio-Dateien verwalten können. Die Titel können der Reihe nach oder in zufälliger Reihenfolge abgespielt werden. Als Quasi-Standard zur Speicherung solcher Listen hat sich das M3U-Format etabliert (`*.m3u`). Das Format solcher Dateien ist denkbar einfach: In jeder Zeile wird einfach ein Dateiname oder eine Internet-Adresse eines MP3-Titels gespeichert. M3U-Dateien waren zwar ursprünglich nur für MP3-Dateien gedacht, werden mittlerweile aber auch für andere Audio-Formate genutzt.

Die meisten Player bieten eine Möglichkeit, ganze Verzeichnisbäume mit Audio-Dateien zu importieren und in M3U-Listen zu speichern. Manchmal kann es aber auch sinnvoll sein, die M3U-Dateien selbst zu erstellen. Das folgende Kommando erstellt eine M3U-Datei für alle MP3-Dateien mit den Dateinamen `/audio/`*gruppe*`/`*titel*`.mp3`:

```
user$ ls /audio/*/*.mp3 ¿ all.m3u
```

Noch allgemeingültiger ist die zweite Variante, die alle MP3-Dateien in `/audio` findet, ganz egal, in welchem Verzeichnis sich die Dateien befinden:

```
user$ find /audio -name '*.mp3' ¿ all.m3u
```

7.3.1 MP3-Format

MP3 steht für *MPEG-1, audio layer 3*, wobei MPEG wiederum eine Abkürzung für *Moving Pictures Experts Group* ist. MP3 ist das bei weitem bekannteste Dateiformat zur Komprimierung von Audio-Daten. Die Verfahren zum Kodieren und Dekodieren von MP3-Dateien sind patentiert. Alle MP3-Encoder müssen also lizenziert werden, selbst dann, wenn sie kostenlos weitergegeben werden sollen. Aus diesem Grund gibt es kaum noch eine Linux-Distribution, die MP3-Encoder direkt mitliefert.

Ein wenig anders sieht die Lage bei MP3-Decodern (MP3-Playern) aus: Obwohl auch diese Technik patentiert ist, haben das Fraunhofer-Institut (die Entwickler des MP3-Formats) und die Firma Thomson zum Ausdruck gebracht, dass Open-Source-Player bis auf weiteres ohne Lizenz eingesetzt werden können. Das ist allerdings keine Garantie für die Zukunft. Die meisten Linux-Distributionen

halten diese etwas vage Zusicherung für ausreichend, um weiterhin MP3-Decoder mit auszuliefern.

Hinweis

Einen sehr guten Überblick über das Thema MP3 und die für Linux verfügbaren MP3-Programme finden Sie im MP3-HOWTO. Wenn Sie sich mehr für die Grundlagen von MP3 interessieren, finden Sie auf den folgenden Websites eine Menge Informationen (auch zu Lizenz- und Patentfragen):
http://www.tldp.org/HOWTO/MP3-HOWTO.html
http://www.iis.fraunhofer.de/amm/
http://www.mpeg.org/MPEG/mp3.html
http://mp3licensing.com/
http://www.mpeg.org/MPEG/mp3-licensing.html
http://www.mp3-tech.org/patents.html

Kommando-orientierte MP3-Player: Ein gemeinsames Merkmal der folgenden Kommandos besteht darin, dass sie keine Benutzeroberfläche zur Verfügung stellen. Stattdessen erfüllen sie nur eine einzige Funktion: Sie dekodieren die als Parameter übergebene MP3-Datei und leiten sie an ein Audio-Device oder in eine WAV-Datei weiter. Die Kommandos sind trotz des fehlenden Komforts wichtig, weil manche anderen Programme darauf zurückgreifen (anstatt selbst einen MP3-Decoder zu implementieren).

- `mpg123` ist das klassische Kommando zum Abspielen von MP3-Dateien. Mit `mpg123 -w out.wav in.mp3` können Sie auch MP3-Dateien in WAV-Dateien umwandeln.
- `madplayer` in eine Alternative zu `mpg123`. Es basiert auf der `libmad`-Bibliothek. Das Programm kann MP3-Dateien in eine ganze Reihe anderer Formate umwandeln. `mad` und `libmad` unterstehen der GPL, was bei `mpg123` nicht der Fall ist.
- `mpg321` ist eine weitere Alternative zu `mpg123`. Wie `madplay` greift es auf `libmad` zurück. Der größte Vorteil von `mpg312` besteht darin, dass man mit `-o` das gewünschte Audio-Device angegeben kann (z. B. `oss`, `alsa`, `arts` oder `esd`).

MP3-Encoder: Wegen der bereits erwähnten Patent- bzw. Lizenzschwierigkeiten gibt es kaum noch Linux-Distributionen, die MP3-Encoder mitliefern. Diverse Encoder sind aber im Internet verfügbar. Die beiden bekanntesten Programme sind `bladeenc` (*http://bladeenc.mp3.no*) und `lame` (*http://www.mp3dev.org/mp3/*).

7.3.2 Ogg-Vorbis-Format

Ogg Vorbis ist die Open-Source-Alternative zum MP3-Format. Ogg Vorbis bringt einen neuen Audio-Datentyp (Dateikennung `.ogg`) samt der Software zum Enkodieren und Dekodieren mit sich. Ogg Vorbis ist als Reaktion auf die Patent- und Lizenzschwierigkeiten mit dem MP3-Format entwickelt worden. Nach Testberichten ist die Audio-Qualität mindestens so gut wie die von MP3. Detaillierte Informationen finden Sie unter *http://www.vorbis.com/*.

Ogg Vorbis ist noch dabei, sich durchzusetzen. Die Verfügbarkeit von Ogg-Vorbis-kompatiblen Audio-Playern nimmt auch zu. Wenn Sie Ihre privaten CDs in Audio-Dateien konvertieren und am Computer anhören möchten, bietet Ogg Vorbis einen perfekten Ausweg aus dem MP3-Lizenzdilemma.

Die `libogg`- und `libvorbis`-Bibliotheken enthalten Kodierungs- und Dekodierungs-Funktionen für Ogg-Vorbis-Bitstreams und -Dateien. Sie werden von manchen Audio-Playern genutzt; bei anderen Playern sind die Decoding-Funktionen direkt ins Programm integriert.

Das Paket `vorbis-tools` enthält unter anderem die Kommandos `oggenc` zum Kodieren, `oggdec` zum Dekodieren und `ogg123` zum Abspielen von Ogg-Dateien. Mit `vorbiscomment` können Sie die Kommentare (Meta-Tags) von Ogg-Dateien lesen und verändern.

7.3.3 Audio-Player

Die Bedienung der hier aufgezählten Programme ist einfach, so dass eine ausführliche Beschreibung hier nur unnötig die Seiten füllen würde. Sehen Sie einfach nach, welches der Programme mit Ihrer Distribution mitgeliefert wird und probieren Sie es aus.

- `mp3blaster` ist ein MP3-Player für den Textmodus. Damit kann das Programm auch als Kommando bzw. ohne X Window System verwendet werden. Das Programm kann M3U-Liedlisten verarbeiten (Option `-l`) und bietet mehr Komfort als `mpg123`.

- `noatun` ist der Standard-Audio-Player des KDE-Systems. Das Programm kommt mit einer ganzen Reihe von Audio-Formaten zurecht. Die Bedienung des Programms ist aber leider nicht sehr intuitiv.

- `rhythmbox` ist ein Gnome-Programm zum Verwalten und Abspielen von Ogg Vorbis- und MP3-Dateien. Die Stärken des Programms liegen in den Verwaltungsfunktionen, dank derer Sie auch in großen Audio-Bibliotheken nicht den Überblick verlieren. Die getestete Version 0.5.4.1 war allerdings noch nicht ganz ausgereift und stürzte mehrfach ab, als ich eine `*.m3u`-Datei zu laden versuchte.

  JuK (ab KDE 3.2) ist ein KDE-Programm mit ähnlichen Funktionen wie `rhythmbox`. Die Bedienung ähnelt der des Apple-Programms iTunes.

- `xmms` ist wahrscheinlich der am weitesten verbreitete Audio-Player für Linux. Das Aussehen von `xmms` lässt sich durch verschiedene Oberflächen (Skins) verändern. Dazu können Winamp-2.0-kompatible Skins verwendet werden. Flexibel ist `xmms` aber auch bei seinen Funktionen, die sich durch Plugins erweitern lassen. Wenn Sie Probleme mit den winzigen Bedienungselementen haben, verdoppeln Sie die Größe des `xmms`-Fensters einfach mit (Strg)+(D). Die weitere Konfiguration erfolgt über das Kontextmenü (rechte Maustaste).

  `xmms` ist zu mehreren Linux-Audio-Systemen kompatibel. Es funktioniert auch wunderfein ohne den Umweg über einen Sound-Server direkt mit dem OSS- oder ALSA-Treiber. Zeigt `xmms` die Fehlermeldung an, dass die Audio-Ausgabe unmöglich sei, wählen Sie im Dialog *Optionen → Einstellungen* (oder (Strg)+(P)) ein anderes Audio-Plugin aus. Weitere Informationen zu `xmms` finden Sie unter *http://xmms.org/*.

- `gmusicbrowser` ist das ideale Tool zur Verwaltung sehr großer MP3- und OGG-Bestände. Das in Perl geschriebene Programm besticht vor allem durch seine Arbeitsgeschwindigkeit. Auch bei großen Musikbeständen spürt der Anwender schnell bestimmte Titel auf oder definiert Filter für bestimmte Interpreten. Ein vollwertiger, integrierter Editor für MP3- und OGG-Tags erlaubt es, die Metadaten einer Musikdatei (alle ID3-Tag-Versionen) zu bearbeiten, ohne den Browser zu verlassen. Zur besseren Eingruppierung von Titeln bietet das Programm fünf Kategorien an. Ein frei definierbares Bewertungssystem rundet das Verwaltungsprogramm ab; zu finden unter *http://squentin.free.fr/gmusicbrowser/gmusicbrowser.html*.

7.3.4 RealPlayer

Die Firma RealNetworks hat eine Reihe kommerzieller Audio- und Video-Formate entwickelt, die in Konkurrenz zum Windows-Media-Format stehen. Zum Abspielen von RealAudio-Dateien ist ein spezieller Player erforderlich. Unter Linux haben Sie die Wahl zwischen den Medien-Playern `xine` und `mplayer` (siehe Seite 299) und dem kommerziellen Programm RealPlayer, das hier vorgestellt wird.

Sie finden das Tool unter *http://forms.real.com/real/player/unix/unix.html* zum kostenlosen Download. Die Version 10 des RealPlayer unterstützt RealAudio, RealVideo 10, MP3, Ogg Vorbis, H263, AAC. Der RealPlayer steht nicht im Quellcode zur Verfügung (kein Open-Source-Projekt). Zur Installation laden Sie das RealPlayer-Setup-Programm (bei unserem Test war dies `RealPlayer10GOLD.bin` von der oben angegebenen Webseite. Diese Datei muss nun ausführbar gemacht (`chmod +x RealPlayer10GOLD.bin`) und dann gestartet werden. Danach läuft der Setup mit dem üblichen Frage-und-Antwort-Spiel ab. Beim ersten Start erfolgt dann noch ein Konfigurations-Dialog mit einem „setup assistant“. Ein Plugin für Mozilla oder einen dazu kompatiblen Webbrowser ist ebenfalls enthalten. Wer will, kann auch ein RPM-Paket herunterladen (`RealPlayer10GOLD.rpm`) und installieren.

Das RealPlayer-Plugin hat den Nachteil, dass es die Dateikennung **.rpm* dem RealPlayer zuordnet. Oft handelt es sich bei `*.rpm`-Dateien aber um RPM-Pakete, die natürlich nicht vom RealPlayer abgespielt, sondern in ein lokales Verzeichnis übertragen werden sollen. Damit das gelingt, müssen Sie in Zukunft RPM-Pakete mit der rechten Maustaste anklicken und *Link-Ziel speichern unter* ausführen.

Die Firma Real engagiert sich stark für das Open-Source-Projekt Helix. Ein Ziel dieses Projekts ist die Entwicklung des so genannten Helix Players, der unter anderem MP3-, Ogg-Vorbis- und RealAudio-Dateien abspielen kann. Weitere Informationen finden Sie unter *https://player.helixcommunity.org/*.

7.3.5 Audio-Konverter

Eine ganze Reihe von Audio-Konvertern wurden in diesem Kapitel bereits vorgestellt. Hier nochmals eine kurze Zusammenfassung:

MP3 → WAV: `mpg123`, `mpg321`, `madplay`
WAV → MP3: `bladeenc`, `lame`
OGG → WAV: `oggdec`
WAV → OGG: `oggenc`

Diese Werkzeuge können auch kombiniert werden, um beispielsweise eine MP3-Datei in das Ogg-Vorbis-Format umzuwandeln. Beachten Sie aber, dass derartige Umwandlungen immer mit Qualitätsverlusten behaftet sind und möglichst vermieden werden sollten! Ein weiterer Nachteil besteht darin, dass die Info-Tags (ID3) verloren gehen. Dieser Nachteil kann durch den Einsatz spezieller MP3-Ogg-Konverter-Scripts umgangen werden (z. B. `mp32ogg` von *http://faceprint.com/code/*).

```
user$ mpg321 -s in.mp3 -w - | oggenc - -o out.ogg
```

SoX steht für *sound exchange* und bietet mit dem Kommando `sox` eine weitere Möglichkeit, Audio-Dateien von einem Format in ein anderes umzuwandeln. `sox` kennt mehr Formate als `sfconvert`. Zu `sox` gibt es die X-Oberfläche `xsox` sowie die Gnome-Oberfläche `gsox`.

7.3.6 CD-Tracks einlesen (Ripper)

Ripper oder **Grabber** lesen Musik-Tracks einer Audio-CD in digitaler Form. Anders als bei einer Daten-CD gibt es aber kein Dateisystem, was das Auslesen komplizierter macht. Bei Lese-Problemen ist es für das CD-Laufwerk schwierig, exakt die Stelle zu finden, an der das Auslesen fortgesetzt werden soll. Ebenso ist es beinahe unmöglich, Fehler auf der CD zu korrigieren. Sowohl die Auslesegeschwindigkeit als auch die Qualität der Audio-Dateien hängt stark von der Qualität des verwendeten CD-Laufwerks ab. Ripper liefern als Ergebnis üblicherweise WAV-Dateien.

Zu den populärsten Linux-Rippern zählen `cdda2wav` und `cdparanoia`, wobei `cdparanoia` den Ruf besitzt, bei zerkratzten CDs und ähnlichen Problemen besonders gute Ergebnisse zu liefern. Beide Kommandos werden mit einer Unzahl von Parametern gesteuert (siehe die jeweiligen Manual-Seiten). Hier nur zwei Beispiele: Das erste Kommando liest Track 3 einer CD im ersten SCSI-CD-Laufwerk. Die resultierende Datei bekommt den Namen `audio.wav`:

```
root# cdda2wav -D /dev/scd0 -t 3
```

Das folgende Kommando liest Track 4 von einer CD im selben Laufwerk. Das Ergebnis ist die Datei `cdda.wav` im lokalen Verzeichnis:

```
root# cdparanoia -d /dev/scd0 "4"
```

Ripper-Benutzeroberflächen: Es wäre recht mühsam, alle Schritte zum Rippen der Titel einer CD manuell auszuführen: Tracks als WAV-Dateien einlesen, umbenennen (CDDB) und schließlich mit einem Encoder in das MP3- oder Ogg-Vorbis-Format umwandeln. Daher wurden Werkzeuge entwickelt, die all diese Aufgaben in einem einzigen Programm vereinen. Diese Programme greifen in der Regel zum Rippen und Kodieren auf externe Programme zurück, stellen also eigentlich nur eine integrative Benutzeroberfläche zur Verfügung.

- Das Gnome-Programm `sound-juicer` ist das komfortabelste Ripper-Werkzeug, das wir gefunden haben. Es verwendet zum Rippen das Programm `cdparanoia`, zum Kodieren per Default `oggenc`.
- Das Gnome-Programm `grip` bietet dieselben Funktionen wie `sound-juicer`, zeichnet sich aber durch eine schier endlose Fülle von Einstellmöglichkeiten aus.
- Auch das Perl-Script `ripit` vereinfacht das Einlesen von Audio-Tracks und deren Umwandlung in MP3- oder Ogg-Vorbis-Dateien – wenn auch ohne grafische Benutzeroberfläche. Vor der ersten Verwendung müssen die Pfade zu den Zusatzprogrammen (Ripper, Encoder etc.) und zum CD-ROM-Device in die ersten Zeilen der Script-Datei eingetragen werden. `ripit` setzt voraus, dass das Perl-Modul `CDDB_get` installiert ist.

7.4 DVDs und Video-Dateien ansehen

Es ist möglich, unter Linux Videos anzusehen, aber der Weg dorthin ist steinig. Video-Dateien enthalten die Video- und Audio-Daten in einem verschlüsselten und komprimierten Format. Es gibt zahlreiche derartige Formate. Zur Dekodierung der Daten gibt es für jedes Format einen so genannten **Codec**, also eine Art Programm-Modul. Die meisten derartigen Codecs stehen für Windows kostenlos zur Verfügung. Den Linux-Markt ignorieren die Entwickler von Video-Formaten

aber in der Regel. Open-Source-Entwickler haben daher viele Codecs neu implementiert. In manchen Fällen war dies aber aus verschiedenen technischen und rechtlichen Gründen nicht möglich.

Abhilfe schuf das `AVIfile`-Projekt: Es hat bewiesen, dass für Windows gedachte Video-Codecs auch unter Linux verwendet werden können. Der erforderliche Aufwand war sogar verhältnismäßig gering. (Das ursprüngliche Projekt bestand aus nur rund 50 kByte Code.) Alle Video- bzw. Medien-Player für Linux greifen mittlerweile auf diese Idee zurück. Die Nutzung von Windows-Codecs ist freilich nicht nur für den Video-Bereich interessant.

Allerdings ist es aus rechtlichen Gründen nicht möglich, für Windows gedachte Codecs als Teil einer Linux-Distribution auszuliefern. Daher müssen Sie die Codecs aus Ihrer Windows-Installation kopieren oder aus dem Internet herunterladen. Die zweite Variante ist oft der einfachere Weg, weil es im Internet Pakete gibt, die fast alle üblichen Codecs enthalten, die frei verfügbar sind.

Anfang 2004 unterstützte `mplayer` immerhin 61 Audio- und 157 Video-Codecs (siehe `/etc/mplayer/codecs.conf`). Dank der vielen Audio-Codecs können Sie mit `xine` und `mplayer` auch Audio-Dateien anhören, die von gewöhnlichen Audio-Playern wie `xmms` nicht unterstützt werden. Dazu zählt beispielsweise das Microsoft-WMA-Format.

7.4.1 Tipps zur Installation und Konfiguration

Bevor die folgenden Abschnitte näher `xine` und `mplayer` beschreiben, finden Sie hier einige allgemeine Tipps zur Installation und Konfiguration.

Zu beiden Programmen finden Sie im Internet vorkompilierte Pakete. Allerdings bereiten die zahlreichen Paketabhängigkeiten oft Probleme. Der Player A benötigt die Bibliotheken B, C und D, die wiederum die Bibliotheken E und F sowie ein Zusatzprogramm G voraussetzen. Auch Paketmanager wie `yum` oder `apt-get`, die solche Abhängigkeiten automatisch lösen sollten, stellen nicht immer die optimale Lösung dar: Oft scheitert die Verwendung dieser Werkzeuge bereits daran, dass Sie keine YUM- oder APT-Datenquellen finden, die alle erforderlichen Pakete enthalten.

Achten Sie darauf, dass es bei manchen vorkompilierten Paketen unterschiedliche Versionen gibt, die optimal an verschiedene Prozessorarchitekturen angepasst sind. Sie benötigen die Version, die zu Ihrer CPU passt (im Zweifelsfall die i586-Version, die auf allen Pentium-kompatiblen CPUs läuft).

Wir können hier nur empfehlen, sich strikt an die Anweisungen auf den jeweiligen Download-Seiten zu halten. Bei uns funktionierte die Installation sowohl von `mplayer` als auch von `xine` auf Anhieb. Eine Alternative zur Installation fertig kompilierter Binärpakete besteht darin, die Programme selbst zu kompilieren.

> **Tipp**
>
> Verwenden Sie auf jeden Fall eine aktuelle Distribution! `xine` und `mplayer` werden in großer Geschwindigkeit weiterentwickelt, d. h. es gibt fast täglich neue Versionen. Die Entwickler verwenden in der Regel aktuelle Compiler- und Bibliotheksversionen und setzen diese voraus. Der Versuch, die Programme auf einer alten Distribution zu installieren, scheitert deshalb oft an alten Bibliotheken.
> Die Dekodierung und Vollbildanzeige von Videos funktioniert nur dann einwandfrei, wenn die CPU des Rechners ausreichend schnell ist, der Zugriff auf das DVD-Laufwerk im DMA-Modus erfolgt und der Video-Player einen optimalen Zugriff auf die Grafikkarte hat.

Bei Geschwindigkeitsproblemen sollten Sie als Erstes den DMA-Modus des DVD-Laufwerks aktivieren, falls dies nicht der Fall ist (`hdparm -d1 /dev/hd`*x*. Das hilft natürlich nur für das Abspielen von DVDs. Auch ein Blick in die X-Konfiguration kann nicht schaden: Insbesondere, wenn die XVideo-Erweiterung nicht zur Verfügung steht, sollten Sie mit einer Farbauflösung von nur 16 Bits per Pixel (bpp) arbeiten, nicht mit 24 oder 32.

7.4.2 xine

Zur Installation von `xine` suchen Sie auf der `xine`-Website die Download-Links. Von dort gelangen Sie zu einer Seite, auf der alle erforderlichen Binärpakete zusammengestellt sind, zum Teil in tagesaktuellen Versionen.

Zum Start des Players führen Sie einfach `xine` aus. Die Steuerung der Grundfunktionen des Programms erfolgt durch ein Bedienfeld. Wenn dieses nicht angezeigt wird, müssen Sie es mit dem `xine`-Kontextmenükommando *Bedienfeld sichtbar* einblenden.

Bevor Sie DVDs abspielen können, müssen Sie zumeist noch den DVD-Device-Namen richtig einstellen. In den Konfigurationsdialog gelangen Sie ebenfalls über das Kontextmenü. Dort geben Sie das gewünschte Device im Dialogblatt *Input* an.

Auch das Navigationsmenü der DVD wird über das `xine`-Kontextmenü eingeblendet. Anschließend können Sie die Menü-Einträge einfach mit der Maus im `xine`-Fenster anklicken. Wenn das nicht funktioniert, führen Sie *Menü → Navigation* aus. Nun erscheint ein recht archaisches Navigationsfeld. Mit dem Kommando `xine-check` können Sie rasch überprüfen, ob alle Voraussetzungen für den optimalen Betrieb von `xine` erfüllt sind. Wenn das nicht der Fall ist, gibt das Kommando Tipps zur Behebung der Mängel.

Neben der Default-Oberfläche (Programm `xine` im Paket `xine-ui`) gibt es eine Menge alternativer Benutzeroberflächen (GUIs), z. B. `gxine` (Gnome) und `kaffeine` (KDE). Auch gibt es ein Mozilla-Plugin zu `xine`, das es ermöglicht, Videos direkt in einem Mozilla-Fenster anzusehen. Nach der Installation des betreffenden Pakets müssen Sie die Datei `/usr/lib/xine-plugin/xineplugin.so` in das Mozilla-Plugin-Verzeichnis kopieren.

7.4.3 mplayer

`mplayer` unterstützt noch mehr Codecs als `xine`, bereitet aber oft auch mehr Probleme bei der Konfiguration und Bedienung. Zur Installation suchen Sie auf der `mplayer`-Website die Download-Links. Von dort gelangen Sie zu einer Seite, auf der alle erforderlichen Binärpakete im RPM-Format zusammengestellt sind. Die Pakete sind nicht nur für Red Hat, sondern auch für andere Distributionen geeignet. Sie benötigen zumindest `mplayer` und `mplayer-common`. Bei unseren Tests waren diese Pakete gegenseitig voneinander abhängig. Zur Installation müssen Sie deshalb beide Pakete gleichzeitig an `rpm` übergeben. Außerdem sollten Sie ein Paket mit den wichtigsten Windows-Codec-Dateien installieren.

Auf der `mplayer`-Website wird das Kompilieren des Quellpakets empfohlen. Es erfolgt mit der üblichen Kommandofolge

```
user$  ./configure
user$  make
user$  make install
```

Die Konfiguration erfolgt über Dateien im `mplayer`-Verzeichnis, in der Regel `/usr/share/mplayer` oder `/etc/mplayer`. In der Datei `codecs.conf` sind alle Codecs und deren Eigenschaften verzeichnet. Man muss die Datei nur ändern, wenn Modifikationen der Eigenschaften nötig sind.

`mplayer` stellt für sich keine Benutzeroberfläche zur Verfügung. Als Parameter übergeben Sie einfach die Video-Datei oder den Device-Namen Ihres DVD-Laufwerks. Falls Ihre Grafikkarte nicht XVideo-kompatibel ist, müssen Sie außerdem die Option `-vo x11` angeben. (Das Programm benötigt dann aber viel mehr Rechenleistung.) Es gibt diverse Tastenkürzel zur Bedienung des Programms (siehe `/etc/mplayer/input.conf`). Die wichtigsten sind (P) (*pause*) und (Q) (*quit*). Mit den Cursortasten bewegen Sie sich im Film vor bzw. zurück.

```
user$  mplayer -vo x11 /dev/sr0
```

Auch zu `mplayer` gibt es eine Benutzeroberfläche (GUI). Die GUI braucht GTK 1.2.x. Die Skins liegen im PNG-Format vor, also werden beim Kompilieren `GTK`, `libpng` sowie `gtk-dev` und `libpng-dev` benötigt. Die GUI kann beim Übersetzen gleich mit erzeugt werden, wenn man den `configure`-Aufruf um den Parameter `--enable-gui` erweitert. Obwohl nicht zwingend vorgeschrieben, sollten die TTF-Fonts für die GUI installiert werden.

Zum Start des Programms verwenden Sie jetzt das Kommando `gmplayer`. Der Player stellt nun eine Konsole für die wichtigsten Funktionen zur Verfügung. Daneben können diverse Einstellungen über das Kontextmenü durchgeführt werden.

Gemäß der Dokumentation (FAQ) sind die aktuellen `mplayer`-Versionen nicht in der Lage, DVD-Navigationsmenüs darzustellen. In der Vergangenheit hat das schon funktioniert, und die Funktion wird möglicherweise in Zukunft wieder

hinzugefügt. Beim Start von `mplayer` können viele Optionen übergeben werden, die in der `man`-Seite ausführlich dokumentiert sind.

Die Optionen können auch in `~/.mplayer/config` bzw. global in `/etc/mplayer/config` eingestellt werden. Jede Option wird in einer eigenen Zeile in der Form *vo=x11* eingetragen. Wenn Sie in `~/.mplayer` eine eigene Datei `codecs.conf` haben, sind nur deren Einträge aktiv, die globale Datei wird ausgeblendet.

7.4.4 Ogle und Videolan

Ogle, ein neuer DVD-Player aus Schweden, wurde auf der Chalmers University of Technology in Göteborg entwickelt und verfügt über etliche interessante Funktionen. Das menügesteuerte Programm besitzt einen sehr guten MPEG2-Decoder mit MMX und SUN-Solaris-mediaLib-Unterstützung. Die Ausgabe des Filmes erfolgt entweder im X11 und XV-Display mit Overlay-Funktion. Ogle unterstützt dank geeigneter Bibliotheken nicht nur freie DVD-Filme, sondern auch CSS-verschlüsselte. Ogle liest nicht nur DVDs, die nicht gemounted sind, sondern auch Daten von der Festplatte.

Unterstützt werden die Audio-Formate AC3, MPEG und LPCM. Eine Einschränkung stellt die Wiedergabe des PCM- und DTS-Audio-Formats dar. Ferner muss das Programm jedesmal neu gestartet werden, um eine neue DVD abzuspielen. Eine Liste der verfügbaren Funktionen finden alle Interessenten auf der Seite des Programmierers (*http://www.dtek.chalmers.se/groups/dvd/*).

Eine weitere Applikation, die das Abspielen von DVDs unter Linux noch einfacher machen will, stellt das Programm `Videolan` dar. Das von Studenten des Pariser Ecole Centrale ins Leben gerufene Projekt setzte sich zum Ziel, einen freien OpenSource DVD-Player für Linux und weitere Plattformen zu entwickeln. Auch hier werden die beiden oben genannten Bibliotheken eingesetzt. Der unter *www.videolan.org* zu findende Player ist für diverse Linux-Distributionen, aber auch für Mac und Windows erhältlich.

7.4.5 CSS

Viele Video-DVDs sind durch **CSS** (*Content Scrambling System*) verschlüsselt. Damit der verschlüsselte Teil der DVD verwertet werden kann, muss die DVD mit einem passenden Schlüssel aufgeschlossen werden. Der dazu nötige Schlüssel ist in lizenzierten Playern enthalten und entstammt einem Pool von etwa 400 Schlüsseln, die von der DVD CCA vergeben werden.

Von Seiten der DVD CCA ist es nur zu verständlich, die Lizenzierung an freie Projekte zu unterbinden, da die Funktionsweise von CSS möglichst geheim gehalten werden soll. Dies ist jedoch mit den Grundsätzen der Open Source Software-Entwicklung unvereinbar. Durch einige Schwachstellen des CSS-Systems gelang es, einen Algorithmus zu entwickeln, der den erforderlichen

Schlüssel für eine DVD „errät“. Code zur Umgehung des CSS (DeCSS) wurde im Internet publiziert und es gibt entsprechende Bibliotheken. In den in diesem Kapitel beschriebenen Programmen ist die Bibliothek in der Regel (je nach Kompilierung) nicht integriert, so dass ein Betrachten von CSS-DVDs ohne die vorherige Installation dieser Zusatzbibliothek unmöglich ist. Inzwischen ist die Umgehung von Kopierschutzverfahren und auch der Hinweis auf dazu geeignete Tools unter Strafe gestellt, weshalb an dieser Stelle auch keine Links aufgeführt sind und Sie auch keine Anleitungen finden.

7.4.6 TV-Anwendungen

Unter Linux gibt es auch etliche TV-Anwendungen unterschiedlichster Art. Das Urgestein ist zweifellos das immer noch aktuelle `xawtv`. Es basiert auf Motif und hat sich im Laufe der Zeit zu einer umfangreichen Suite entwickelt, die das Hauptprogramm `xawtv` umschließt. `xawtv` kann Videos (unkomprimiert) und durch ein Plugin sogar Quicktime aufnehmen. Leider verfügt `xawtv` nur über eine etwas archaische GUI.

Völlig ohne GUI kommt **Tvtime** aus. Hier wird besonders auf die Qualität des Fernsehbildes geachtet. Trotz der fehlenden GUI lässt sich `tvtime` intuitiv und komfortabel bedienen, nicht zuletzt durch das eingebaute OSD. Dort können auch die Einstellungen zur Bildausgabe vorgenommen werden (z. B. Auflösung, Framerate, Farbe, Helligkeit, Kontrast etc.).

Die Window-Manager KDE und Gnome bringen jeweils ihre eigenen Fernsehanwendungen mit, **kdetv** (`kwintv` neu geschrieben) und **zapping** (Gnome). Beide unterscheiden sich in Funktionalität und Bedienbarkeit kaum voneinander. `kdetv` besitzt eine Plugin-Schnittstelle, die den Spielraum für neue Funktionen erweitert. Durch Plugins können u. a. ein OSD und Videotext dargestellt und die Ein- und Ausgabe verändert werden. `zapping` bietet die gleichen Features wie `kdetv`, kann jedoch durch Plugins auch Videos aufnehmen.

Einen anderen Weg geht **Freevo**. Es handelt sich um ein Multimedia-Center, das optisch an eine Set-Top-Box erinnert. Die einfache intuitive Bedienung und LIRC-Unterstützung lassen `freevo` als Lösung für einen Wohnzimmer-PC ideal erscheinen. `freevo` kann auch noch aufnehmen, Bilder auf der Festplatte betrachten und vieles mehr (Plugins).

`freevo` benötigt eine Python-Umgebung. Für verschiedene Distributionen stehen RPM-Dateien zur Verfügung, die den Umgang mit den zahlreichen Abhängigkeiten erleichtern. Während der Einrichtung wählt man über eine Konsolenoberfläche die nötigsten Einstellungen (TV-Norm, Kanalliste, Auflösung) und kann danach `freevo` starten.

Sehr ähnlich zu `freevo` ist **MythTV**, ein anderes Multimedia-Center-Projekt. Es bietet Support für mehrere TV-Karten (Bild-im-Bild), Timeshifting, Analog-, Digital-TV-Karten sowie PVR-Karten. Zahlreiche Plugins lassen kaum Wünsche offen. Die Installation von MythTV ist kompliziert. Es wird ein laufender

MySQL-Server benötigt (zur Speicherung der Programminformationen) sowie einige andere Abhängigkeiten. Hat man die Grundvoraussetzungen einmal erfüllt, startet man als `root` die Kommandos `mythfilldatabase` zum Füllen der MySQL-Datenbank) und `mythbackend` für die Verbindung zur Datenbank usw. Anschließend werden mit `mythfrontend` die letzten Einstellungen vorgenommen. Die Downloads und eine detaillierte Installationshilfe finden sich unter *http://www.mythtv.org/*.

7.5 Digitalkameras, Scanner, Bildverarbeitung

Dieser Abschnitt beschreibt, wie Sie unter Linux Bilder verarbeiten, die Sie zuvor mit einer Digitalkamera aufgenommen bzw. mit einem Scanner eingelesen haben. Im Detail werden die Übertragung von Bildern von einer Digitalkamera auf den Linux-Rechner und die Konfiguration und Bedienung eines Scanners behandelt. Ausserdem erhalten Sie eine Übersicht über Tools zur Erstellung von Screenshots (also Abbildungen des Bildschirms bzw. eines einzelnen Fensters) und wir behandeln kurz einige Programme zur Bildverarbeitung.

Fast alle Digitalkameras sehen für die Übertragung der Bilder zum PC ein USB-Kabel vor. Uneins sind sich die Kamerahersteller allerdings über das Protokoll zum Datenaustausch:

- **USB-Datenträger:** Am einfachsten ist der Datentransport bei Kameras, die sich wie ein USB-Datenträger verhalten. In diesem Fall behandelt Linux die Kamera wie eine Festplatte mit USB-Anschluss bzw. wie einen USB-Memory-Stick.
- **PTP-Kameras:** Die meisten neueren Kameras unterstützen das *picture transfer protocol* (PTP). Dieses Protokoll sieht nicht nur Kommandos zur Übertragung von Bildern vor, sondern auch einfache Steuerungsfunktionen (z. B. um per Computersteuerung ein Foto zu erstellen).
- **Kameras mit herstellerspezifischem Protokoll:** Schließlich gibt es eine Menge zumeist älterer Modelle, die nur ein herstellerspezifisches Protokoll unterstützen. Von diesen Kameras werden unter Linux zwar die meisten populären Modelle unterstützt, aber leider nicht alle.

Zur Übertragung der Bilder stehen je nach Protokoll verschiedene Linux-Programme zur Auswahl: Für Kameras des ersten Typs kann jeder beliebige Datei-Manager eingesetzt werden. Allerdings muss die Kamera vorher in das Dateisystem eingebunden werden. Bei manchen Distributionen erscheint dazu sogar automatisch beim Anstecken der Kamera ein Icon auf dem Desktop (automount). Ist das nicht der Fall, müssen Sie die Kamera als USB-Datenträger behandeln.

Für die Kameras der anderen beiden Typen stellt die Bibliothek `gphoto2` Kommunikationsfunktionen zur Verfügung. Die direkte Anwendung dieser Funktionen ist recht umständlich, glücklicherweise gibt es aber sowohl für KDE als auch für Gnome komfortable Oberflächen: `digikam` und `gtkam` sowie `gThumb`. Die Programme werden im Folgenden kurz beschrieben. Auch der KDE-Universal-Browser Konqueror ist in der Lage, das PTP-Protokoll korrekt zu interpretieren. Dazu geben Sie einfach als Adresse `camera:/` an.

Statt der Digitalkamera können Sie auch deren Speicherkarte mit dem Computer verbinden. Dazu benötigen Sie zwar ein zusätzliches, billiges Speicherkartenlesegerät, aber diese Vorgehensweise hat auch zwei Vorteile: Die Datenübertragung funktioniert unabhängig vom Protokoll der Kamera und beansprucht die Batterie der Kamera nicht.

7.5.1 digikam (KDE)

`digikam` unterstützt Kameras aller drei oben genannten Typen. Beim ersten Start des Programms muss allerdings eine Konfiguration erfolgen. In den meisten Fällen ist es ausreichend, im Dialog *Configure* → *Setup* den Button *Auto-Detect* anzuklicken. `digikam` unterstützt auch Kameras, die sich wie ein USB-Speichermedium verhalten. Allerdings muss die Kamera in diesem Fall vorher in das Dateisystem integriert werden (`mount`). Außerdem ist eine manuelle Konfiguration erforderlich, bei der Sie als Kameratyp *USB mass storage* und das `mount`-Verzeichnis angeben.

Das Programmfenster setzt sich aus drei Teilen zusammen: aus der Albumliste links oben, aus dem Kameraverzeichnisbaum links unten und aus der Bildvorschau rechts. Bei der Albumliste handelt es sich um ganz gewöhnliche Verzeichnisse innerhalb des Home-Verzeichnisses. Wenn ein Verzeichnis angeklickt wird, zeigt das Programm alle enthaltenen Bilder an.

Klicken Sie auf das Kamerasymbol links unten, werden alle auf der Kamera befindlichen Bilder angezeigt. Dabei wird ein anderer Hintergrund verwendet, damit klar ist, dass es sich nun um Kamerabilder handelt. Im Vorschaubereich können Sie die gewünschten Bilder markieren und dann per Kontextmenü in das aktuelle Verzeichnis oder per Drag&Drop in ein beliebiges Verzeichnis kopieren. Nach dem Kopieren können die Bilder gelöscht werden.

7.5.2 gtkam (Gnome)

`gtkam` ist das Gnome-Gegenstück zu `digikam`. Mit Ausnahme des USB-Massenspeicher-Typs unterstützt es dieselben Kameras wie `digikam`. Vor der ersten Verwendung muss die Digitalkamera konfiguriert werden: Dazu führen Sie einfach *Kamera* → *Kamera hinzufügen* aus und klicken die Buttons *Erkennung* und dann *OK* an.

Das Programm zeigt links einen Baum an, der die interne Verzeichnisstruktur der Kamera wiedergibt. In diesem Baum können Sie beliebig viele Zweige (Verzeichnisse) gleichzeitig per Mausklick auswählen. Im rechten Fensterbereich werden Vorschausymbole aller Bilder aller markierten Verzeichnisse angezeigt. Nun markieren Sie per Mausklick die Bilder, die Sie auf Ihren Rechner übertragen möchten. (*Auswählen* → *Alle* markiert alle Bilder.)

Die eigentliche Übertragung erfolgt durch *Datei* → *Speichere ausgewählte Photos*, wobei ein Dialog zur Auswahl des Zielverzeichnisses erscheint. Anschließend können die Bilder mit *Datei* → *Lösche Photos* auf der Kamera gelöscht werden.

7.5.3 gThumb (Gnome)

`gThumb` (*gthumb.sourceforge.net/*) scheint `gtkam` mittlerweile abzulösen. Es ist ebenfalls ein Programm, mit dem man Bilder betrachten, organisieren und bearbeiten kann. Neben den Funktionen als Image-Viewer und -Browser (Bildformate BMP, JPEG, GIF, PNG, TIFF, ICO, XPM) erlaubt es die Anzeige von EXIF-Daten und die Organisation von Bildern in Katalogen. Eine komfortable Suchfunktion hilft beim Ordnen der Bilder.

`gThumb` ermöglicht einfach Bildbearbeitungsfunktionen, z. B. Drehen, Ändern des Dateityps, Ändern und Anpassen der Farben sowie das Hinzufügen von Kommentaren zu Bildern. Bei der Anzeige kann man zwischen Fenster- und Vollbildanzeige wechseln. Außerdem lassen sich Diashows und Webalben erstellen .

`gThumb` durchsucht nach dem Start Ihr home-Verzeichnis. Sind in diesem Verzeichnis Bilder vorhanden, generiert das Gallery-Panel automatisch Thumbnails, die Sie markieren und im Hauptanzeigebereich anzeigen können. Wenn Sie auf ein Thumbnail doppelklicken, wird dies im Hauptfenster angezeigt. Sie können das Bild hinein- und herauszoomen, einen Vollbildschirm einstellen und auf einem konfigurierten Drucker ausdrucken. Über die Toolbar lassen sich das Bild an das Anzeigefenster anpassen, mehrere Dateien als Katalog für leichteren Zugang zusammenfassen und Beschreibungen zu den Bildern erstellen.

Die `gThumb`-Schnittstelle besitzt außerdem ein Textfeld, in das Sie einen bestimmten Pfad zu den Bildverzeichnissen eingeben können. Drücken Sie die rechte Maustaste im Anzeigefenster, öffnet sich ein PoUp-Fenster mit Dateioptionen, wie zum Beispiel Umbenennen, Verschieben, Kopieren und Bilder von einem Dateiformat in ein anderes umwandeln.

Markieren Sie das erste Bild in der Dateiliste der Miniaturbilder links. Wenn Sie dann auf „Diashow“ in der Toolbar klicken, wird eine Vollbild-Diashow gestartet. Jedes Bild wird ca. vier Sekunden lang angezeigt. Sie können die Diashow jederzeit durch Drücken der (Esc)-Taste beenden.

Sie können `gThumb` konfigurieren, indem Sie *Bearbeiten* → *Präferenzen* wählen. Es lassen sich das Layout des Applikationsfensters ändern, ein Standard-

Bilderverzeichnis beim Starten einstellen, die Größe von Thumbnails ändern und das Intervall der Bildanzeige bei Diashows wählen.

7.5.4 gphoto2

`digikam` und `gtkam` sind Benutzeroberflächen zur `gphoto2`-Bibliothek. Die Funktionen dieser Bibliothek können Sie auch durch das gleichnamige Kommando nutzen. Beispielsweise versucht `gphoto2 --auto-detect`, die angeschlossene Kamera zu erkennen, und zeigt die entsprechenden Informationen an. `gphoto2 --get-all-thumbnails` überträgt verkleinerte Symbole aller auf der Kamera gespeicherten Bilder in das lokale Verzeichnis. Bemerkenswert ist, dass bei manchen Kameras mit `gphoto2 --capture-image` sogar per Kommando Fotos erstellt werden können, so dass die Kamera wie eine Webcam genutzt oder für Überwachungszwecke eingesetzt werden kann. Umfassende Informationen zu `gphoto2` finden Sie auf der Website *http://gphoto.sourceforge.net/*.

7.5.5 Lesegeräte für Speicherkarten

Anstatt die Kamera direkt mit dem Computer zu verbinden, können Sie auch eine direkte Kommunikation mit der Speicherkarte der Kamera herstellen. Dazu gibt es kostengünstige interne und externe Kartenlesegeräte, die üblicherweise per USB-Kabel mit dem Computer verbunden werden. Eine attraktive Variante für Notebook-Besitzer sind winzige PCMCIA-Adapter, in die die Speicherkarte eingesteckt werden kann.

Speicherkartenlesegeräte werden zumeist wie eine zusätzliche SCSI-Festplatte behandelt (so wie Windows eigene Laufwerke dafür „produziert"). Eine Ausnahme von dieser SCSI-Regel können PCMCIA-Adapter für CompactFlash-Karten bilden, die sich als neue IDE-Schnittstelle identifizieren (z. B. `/dev/hde`). Bei Geräten für verschiedene Speichermedien gelten je nach Speicherkartentyp unterschiedliche Schnittstellennamen – z. B. `/dev/sda1` für CompactFlash-Karten oder `/dev/sda2` für Memory-Sticks. Für den Datenzugriff gelten dieselben Regeln wie für den Umgang mit USB-Festplatten, USB-Memory-Sticks etc. Obwohl die diesbezüglichen Linux-Treiber als recht ausgereift gelten, können manchmal Probleme auftreten, beispielsweise mit verschiedenen Versionen des USB-Moduls oder der Wechsel der Speicherkarte (natürlich mit vorherigem `umount` und anschließendem `mount`) funktioniert erst, nachdem das Lesegerät aus- und wieder angesteckt wurde.

7.5.6 Scanner

Die zurzeit erhältlichen Scanner werden üblicherweise an eine der drei folgenden Schnittstellen angeschlossen: USB, SCSI oder parallele Schnittstelle. Grundsätzlich kommt Linux mit allen drei Typen zurecht, es hängt vielmehr vom individuellen Modell ab, ob Linux damit umgehen kann

oder nicht. Erkundigen Sie sich unbedingt vor dem Scanner-Kauf, ob das Gerät Linux-kompatibel ist, z. B. bei *http://panda.mostang.com/sane/sane-backends.html*, *http://www.buzzard.org.uk/jonathan/scanners-usb.html* oder *http://www.buzzard.org.uk/jonathan/scanners.html*.

Scanner-Konfiguration

Für den Scanner-Zugriff ist das Programmpaket SANE verantwortlich (*Scanner Access Now Easy*). In manchen Fällen wird Ihr Scanner sofort beim Anstecken erkannt, d. h. die ab Seite 310 beschriebenen Scan-Programme funktionieren auf Anhieb. Im Regelfall müssen Sie SANE aber zuerst konfigurieren, bevor Sie Ihren Scanner einsetzen können.

Bei einigen Distributionen gibt es dazu eigene Konfigurationswerkzeuge, etwa bei Mandriva (`drakconf`) oder Suse (YaST-Modul *Hardware* → *Scanner*). Wenn ein derartiges Konfigurationswerkzeug fehlt, müssen Sie die Konfigurationsdateien im Verzeichnis `/etc/sane.d/*` selbst modifizieren. Die erforderlichen Schritte sind im Folgenden für USB- und SCSI-Geräte beschrieben.

Die Verwendung eines **USB-Scanners** setzt voraus, dass die USB-Basismodule sowie das Modul *scanner* geladen sind. Das erfolgt in der Regel automatisch. Sie können sich davon mit `lsmod` überzeugen bzw. das `scanner`-Modul gegebenenfalls manuell mit `modprobe` manuell laden.

Der zweite Schritt besteht darin, die Produktdatei im Verzeichnis `/etc/sane.d` so zu ändern, dass SANE den Scanner findet. Normalerweise sind diese Dateien so eingerichtet, dass auf das Device `/dev/scanner` zugegriffen wird. Für USB-Scanner muss stattdessen aber `/dev/usbscanner` oder `/dev/usb/scanner0` verwendet werden.

Suchen Sie die Produktdatei Ihres Scanners, beispielsweise `/etc/sane.d/epson.conf`, und ändern Sie darin die Gerätezuordnung, damit der Scanner korrekt erkannt wird. (Da die SANE-Konfigurationsdateien normalerweise für SCSI-Scanner vorkonfiguriert sind, müssen Sie die Änderung selbst vornehmen.):

```
# /etc/sane.d/epson.conf
usb  /dev/usb/scanner0
```

`scanimage -L` sollte den Scanner jetzt selbstständig erkennen:

```
root# scanimage -L
device 'epson:/dev/usb/scanner0' is a Epson Perfection1240 flatbed scanner
```

SANE berücksichtigt nur die Produktdateien, die in `/etc/sane.d/dll.conf` angegeben sind. Wenn SANE Ihren Scanner nicht erkennt und Sie sich vergewissert haben, dass SANE diesen Scanner prinzipiell unterstützt, sollten Sie einen Blick in `dll.conf` werfen und sicherstellen, dass der Produktname Ihres Scanners dort nicht auskommentiert ist.

SCSI-Scanner werden mit dem SCSI-Generic-Treiber (Modul `sg`) angesprochen. Neben diesem Modul müssen außerdem ein Modul mit den SCSI-Grundfunktionen (`scsi_mod`) sowie ein Modul mit dem SCSI-Lowlevel-Treiber für die SCSI-Karte geladen sein. Falls das nicht automatisch der Fall ist, müssen Sie die drei Module mit `modprobe` laden. Bitte beachten Sie, dass der Scanner bereits vor dem Laden der Module eingeschaltet werden muss!

SANE spricht den Scanner wahlweise über `/dev/scanner` oder über `/dev/sg`*x* an, wobei das Programm selbst in der Lage ist, die richtige Device-Datei zu finden. Dazu muss die SANE-Gerätedatei (also z. B. `/etc/sane.d/epson.conf`) die folgende Zeile enthalten. (Das ist meistens bereits per Default der Fall, so dass keine Änderung notwendig ist.)

```
# /etc/sane.d/epson.conf
scsi epson
```

Scannen mit SANE und Xsane

Nach der Konfiguration von SANE stehen Ihnen verschiedene Programme zum Scannen von Bildern zur Auswahl. Diese Programme werden in den folgenden Abschnitten kurz vorgestellt. Zwei nicht besonders bedienungsfreundliche Scan-Programme sind Teil des SANE-Pakets und stehen immer zur Verfügung, `scanimage`, ein textorientiertes Kommando zum Scannen und `xscanimage`, eine komfortable X-Oberfläche zum Scannen.

Bei `xscanimage` wird das Bild zuerst in einem Preview-Fenster in niedriger Auflösung gescannt. Anschließend wird dort der relevante Ausschnitt markiert, der anschließend in höherer Qualität gescannt und direkt in eine Datei geschrieben wird. Dabei werden je nach Modus (Schwarz-Weiß, Graustufen, Farben) die Bitmap-Formate PBM, PGM oder PPM verwendet. Die meisten Linux-Programme zur Bildverarbeitung kommen mit diesen Formaten zurecht (aber leider nicht alle).

`xscanimage` kann mit dem Menükommando *Xtns* → *Aquire Image* auch direkt von Gimp aus gestartet werden. Das hat den Vorteil, dass das Bild sofort mit Gimp verarbeitet und dann in einem beliebigen von Gimp unterstützten Format gespeichert werden kann.

Das Programm `xsane` bietet erheblich mehr Konfigurationsmöglichkeiten (Farb- und Kontrastkorrektur etc.) als die SANE-Scan-Programme. Das Programm nutzt wie alle anderen Linux-Scan-Programme die SANE-Bibliothek, ist aber nicht Teil des SANE-Pakets und muss daher extra installiert werden. Die wichtigsten Vorteile gegenüber `xscanimage` sind:

- Die Auswirkungen einer Farbkorrektur können bereits im Preview-Fenster betrachtet werden.

- Im Preview-Fenster können mit so genannten Pipetten Punkte markiert werden, die weiß oder schwarz sind. Auf diese Weise kann das Programm einen Weißabgleich durchführen.
- Jedes eingescannte Bild erscheint in einem eigenen Fenster und kann dann in verschiedenen Dateiformaten gespeichert werden (z. B. JPEG, PNG, EPS).
- Für das eingescannte Bild kann eine Texterkennung durchgeführt werden. Dazu muss auch das Programm `gocr` installiert sein. Der erkannte Text wird in einer eigenen Datei gespeichert (per Default `out.txt`). Meine Tests dieser Funktion verliefen allerdings eher enttäuschend.
- `xsane` kann nach vorheriger Konfiguration auch dazu verwendet werden, das eingescannte Bild sofort auszudrucken (Copy-Funktion) oder zu faxen.

Das KDE-Programm `kooka` bietet zwar weniger Funktionen als `xsane`, ist dafür aber etwas einfacher zu bedienen. Alle eingescannten Bilder werden automatisch gespeichert (in `.kde/share/apps/ScanImages/`).

7.5.7 Screenshots erstellen

Ein Screenshot ist ein Abbild des aktuellen Bildschirm- oder Fensterinhalts in einer Grafikdatei. Wir stellen Ihnen vier Programme dazu vor:

- **ksnapshot (KDE):**
  Die Bedienung des Programms ist einfach: Der Button *Neues Bildschirmfoto* startet die Screenshot-Funktion. Das Programm `ksnapshot` wird nun unsichtbar. Nach der zuvor eingestellten Zeit (per Default zehn Sekunden) wird wahlweise der gesamte Bildschirminhalt oder nur das Fenster unter dem Mauscursor kopiert.

  Das resultierende Bild wird nicht automatisch gespeichert – dazu müssen Sie *Speichern* anklicken. Beim Speichern wird je nach Dateikennung automatisch das entsprechende Format verwendet. PostScript-Dateien erhalten Sie nur mit der Kennung `.eps` (nicht mit `*.ps`) – und es stimmt die BoundingBox nicht. Abhilfe: Verwenden Sie zum Speichern ein Bitmap-Format und wandeln Sie die resultierende Datei anschließend mit einem anderen Programm in das PostScript-Format um!
- **gnome-panel-screenshot (Gnome)**
  Bei Gnome ist das Programm `gnome-panel-screenshot` direkt in den Desktop integriert. Wie unter Windows erstellt (Druck) einen Screenshot des gesamten Bildschirms, (Alt)+(Druck) eine Abbildung des gerade aktiven Fensters. Anschließend wird das Programmfenster sichtbar und bietet die Möglichkeit, die Abbildung im PNG-Format zu speichern. (Alt)+(Druck) hat leider den Nachteil, dass das Bild von einem schwarzen Rahmen in der Breite

eines Pixels umgeben wird. Die einzige Abhilfe besteht darin, den Screenshot mit (Druck) bei einem weißem Desktop-Hintergrund zu erstellen und den relevanten Bildbereich anschließend mit einem anderen Programm auszuschneiden.

- **gimp**
  Auch das Bildverarbeitungsprogramm Gimp weist eine einfache Screenshot-Funktion auf. Sie wird mit dem (Kommando *Xtns* → *Screenshot* gestartet.
- **xv**
  Das Programm `xv` kann ebenfalls zur Erstellung von Screenshots eingesetzt werden. Dazu klicken Sie zuerst den Button *Grab* und anschließend ein Fenster oder den Hintergrund des X-Systems an. `xv` liest daraufhin den Inhalt des Fensters oder des ganzen Bildschirms ein. Anschließend kann das Bild bearbeitet und gespeichert werden.

Tipp

Wenn Sie in einem Screenshot mehrere Fenster darstellen möchten, sollten Sie vorher den Bildschirmhintergrund auf Weiß stellen, damit dieser nicht stört. Unter KDE und Gnome verwenden Sie dazu das Kontrollzentrum. Falls Sie ohne ein Desktop-System unter X arbeiten, führt das Kommando `xsetroot -solid white` zum gewünschten Ergebnis. Ist das Screenshot-Programm nicht in der Lage, den gewünschten Bereich einzugrenzen, erstellen Sie einfach einen Screenshot des gesamten Bildschirms. Anschließend können Sie ein Bildverarbeitungsprogramm dazu verwenden, den tatsächlich benötigten Ausschnitt zu extrahieren.

7.5.8 Bildbetrachtung und -konvertierung

Nachdem die ersten Abschnitte dieses Kapitels verschiedene Möglichkeiten zum Einlesen bzw. Erzeugen von Bildern vorgestellt haben, geht es nun darum, diese Bilder effizient anzusehen, zu ordnen und weiterzuverarbeiten. Dazu stehen unter Linux zahlreiche Programme zur Verfügung, von denen hier die wichtigsten kurz vorgestellt werden.

- Der KDE-Browser Konqueror verfügt über eine sehr leistungsfähige und schnelle Bildvorschau. Sie wird durch *Ansicht* → *Anzeigemodus* → *Symbolmodus* aktiviert. Die Größe der Symbole stellen Sie mit *Ansicht* → *Symbolgröße* ein. Aus Geschwindigkeitsgründen wird die Vorschau bei sehr großen Dateien deaktiviert. Der Grenzwert (Default 1 MByte) kann im Dialog *Einstellungen* → *Konqueror einrichten* → *Dateibetrachter* verändert werden.

  Wenn bei Bildern von Digitalkameras die Orientierung (Hoch- oder Querformat) gespeichert wurde, werden diese Bilder automatisch korrekt gedreht. Außerdem zeigt das Programm diverse Zusatzinformationen zu dem Bild an,

über dem sich die Maus gerade befindet. Noch mehr Informationen liefert das Kontextmenükommando *Eigenschaften*.

Damit nicht jedes Mal alle Bilder des Verzeichnisses neu eingelesen werden müssen, verwaltet Konqueror einen Cache mit den verkleinerten Symbolen aller Bilder (Verzeichnis `~/.kde/share/thumbnails`).

- `kuickshow` ist ein KDE-Bildbetrachter. Ein Kontextmenü bzw. entsprechende Tastenkürzel führen zu einigen Kommandos für einfache Bildbearbeitungsfunktionen (Vergrößern, Verkleinern, Drehen, Kippen, Helligkeit ändern, Speichern etc.). Aber vor allem kann man schnell und komfortabel (per Mausrad!) durch alle Bilder eines Verzeichnisses blättern. Wird beim Start von `kuickshow` der Name einer Bilddatei übergeben (bzw. das Programm vom Konqueror aus gestartet), stehen nur die o. a. Funktionen zur Verfügung. Starten Sie dagegen das Programm über das KDE-Menü oder per Kommando, erscheint ein eigenes Browser-Fenster, aus dem heraus verschiedene Voreinstellungen verändert (Vollbildmodus) und eine Diashow gestartet werden kann.

- Auch der Gnome-Datei-Manager kann Bilddateien in einer verkleinerten Symbolansicht anzeigen. Allerdings erscheinen die Symbole mit zunehmender Größe unscharf, weil es sich hierbei aus Geschwindigkeitsgründen nicht um Verkleinerungen der Originalbilder, sondern um Vergrößerungen der erstmalig erzeugten Miniaturansichten handelt.

  Die Bildvorschau funktioniert per Default allerdings nur bei lokalen Dateien, nicht aber, wenn die Dateien über ein Netzwerkverzeichnis (z. B. per NFS) zugänglich sind. Wenn Sie eine Vorschau auch in diesem Fall wünschen, aktivieren Sie im Dialog *Bearbeiten* → *Einstellungen* → *Vorschau* die Einstellung *Miniaturvorschaubilder immer anzeigen*. Im selben Dialog können Sie auch einstellen, bis zu welcher Dateigröße die Vorschau durchgeführt wird (Default 3 MByte).

  Damit die Bilder nach einem Verzeichniswechsel nicht wieder neu eingelesen werden müssen, verwaltet Nautilus einen Cache mit den verkleinerten Symbolen aller bisher betrachteten Bilder (Verzeichnis `~/.nautilus/thumbnails`).

- Auch `gthumb` liefert einen raschen Überblick über die Bilder eines Verzeichnisses. Die Stärken dieses Programms liegen aber in der systematischen Verwaltung der Bilder: Die Bilder können kommentiert, kategorisiert und in Kataloge eingefügt werden. Nach diesen Zusatzinformationen kann anschließend auch gesucht werden.

  `gthumb` erkennt die automatische Orientierung von Digitalkamerabildern leider nicht, bietet dafür aber eine komfortable Möglichkeit, die Bilder zu drehen. Dabei kommt es bei komprimierten Bildformaten wie JPEG zu keinem Qualitätsverlust (der sonst bei jedem Speichervorgang wegen der neuerlichen Komprimierung eintritt).

Darüber hinaus unterstützt `gthumb` einige elementare Bildverarbeitungsfunktionen. Diese Funktionen verändern nur das im Fenster links unten angezeigte Bild, nicht aber die zugrunde liegende Datei. Wenn Sie die Änderungen speichern möchten, müssen Sie dies explizit tun (*Datei* → *Speichern unter*).

- Das Programm `xv` ist ein altgedientes Unix-Programm (erstes Copyright 1989). Die Bedienung erinnert auch noch an frühe Window-Manager. Es bietet aber einige Funktionen zur Bild- und Farbbearbeitung, die die Einarbeitung durchaus rechtfertigen (z. B. Formatkonvertierungen). Mit `xv` können auch Screenshots erstellt werden. Nach dem Start erscheint ein Fenster mit einem bunten Logo oder dem als Dateinamen übergebenen Bild. Das Steuerungsfenster von `xv` erscheint erst nach einem Klick mit der rechten Maustaste. Anschließend können Sie einige elementare Bearbeitungsschritte direkt über Buttons des Steuerungsfensters ausführen. Beispielsweise stellt das Menü *Algorithms* Ihnen diverse Algorithmen zur Bildverarbeitung zur Verfügung.

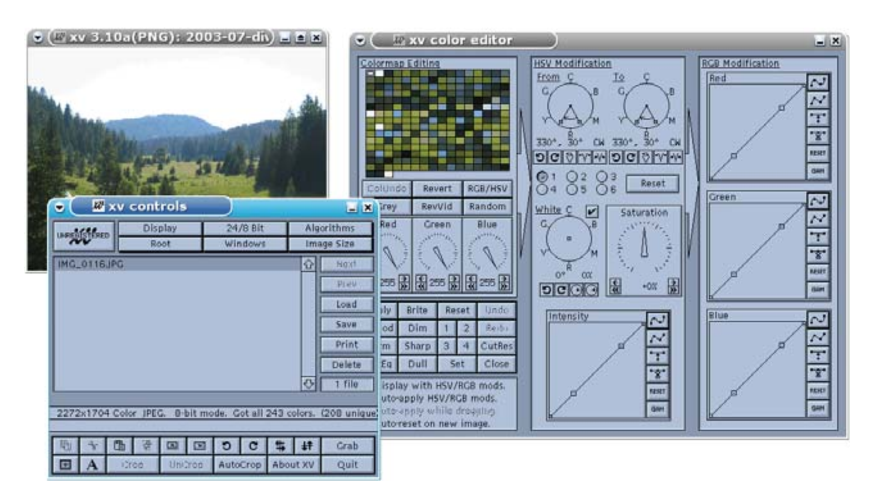


Abbildung 7.1: Bildbearbeitung mit xv

Helligkeit, Kontrast und andere Farbeinstellungen ändern Sie mit *Windows* → *Color Editor* in einem eigenen Fenster. Im 256-Farben-Modus können Sie jede einzelne Farbe getrennt verändern. Die gewünschte Farbe kann mit der mittleren Maustaste direkt am Bild ausgewählt werden. Bei Bildern in Echtfarben besteht diese Möglichkeit nicht, Sie können aber mit *24/8 Bit* → *8 Bit Mode* das Bild auf 256 Farben reduzieren. Vorher sollten Sie mit *24/8 Bit* → *Best 24* → *8* dafür sorgen, dass dabei eine optimale Bildqualität gewährleistet ist.

Mit der linken Maustaste können Sie einen beliebigen Bildausschnitt markieren und anschließend mit *Crop* das Bild entsprechend verkleinern. Damit

haben Sie die Möglichkeit, nur einen Teil des Bilds zu speichern. (Alle Bildverarbeitungsschritte werden für das gesamte Bild durchgeführt. Wenn Sie *Uncrop* ausführen, wird wieder das gesamte Bild angezeigt.)

Um mit `xv` möglichst kompakte PostScript-Dateien zu erzeugen, wandeln Sie die Grafiken vorher in den 8-Bit-Farbmodus um. Anschließend wählen Sie im PostScript-Speicherdialog die Option *Compress* aus.

- Falls Sie Bilder von einer Foto-CD lesen möchten, empfiehlt sich dazu das Programm `xpcd`. Die CD muss vorher ganz normal mit `mount` in das Dateisystem eingebunden werden. xpcd <cdrom-verzeichnis> zeigt dann ein Inhaltsverzeichnis der CD an. Anschließend können Sie die einzelnen Bilder in unterschiedlichen Auflösungen ansehen und dann in den Formaten JPEG, PPM oder TIFF speichern.

- **Image Magick:** Das Programmpaket Image Magick stellt mehrere Einzelkommandos zur Verfügung, um Bilddateien zu bearbeiten (z. B. Vergrößern, Verkleinern oder Konvertieren). Zusammen mit dem Paket wird eine umfassende Beschreibung aller Kommandos im HTML-Format installiert. Für Programmierer steht Image Magick auch als Bibliothek zur Verfügung. Wenn Sie sich interaktiv einen Überblick über die Fähigkeiten des Pakets verschaffen möchten, können Sie mit `display dateiname` die eher altmodische Benutzeroberfläche starten. Die Stärken des Programms liegen eher darin, ganze Verzeichnisse von Bildern automatisiert zu verarbeiten. Der folgende Befehl speichert die Datei `picture.jpg` im Format 200*200 Pixel als `pic-small.jpg` ab, wobei die Proportionen des Originalbilds erhalten bleiben. Bilder, die kleiner sind, bleiben unverändert. Für die Verkleinerung ist die Option `-resize` verantwortlich. `-size` bewirkt lediglich eine schnellere Verarbeitung:

```
convert -size 400x400 \
        -resize 400x400 \
        picture.jpg pic-small.jpg
```

Wenn Sie mit keinem der hier beschriebenen Programme zufrieden sind, sollten Sie auch `pixie`, `kview`, `kimdaba`, `gtksee` und `gqview` ausprobieren, die aus Platzgründen hier nicht beschrieben werden. An dieser Stelle fehlt auch die Beschreibung von Gimp, das an anderer Stelle in diesem Buch behandelt wird.

7.6 CDs und DVDs brennen

Wie Sie sicher wissen, gibt es bei beschreibbaren CDs und DVDs eine verwirrende Typenvielfalt. Begonnen hat alles mit der **CD-ROM**, der ursprünglichen *Read-Only*-CD. Dann kam die **CD-R** auf den Markt, die nur einmal beschrieben werden kann. Die Multi-Session-Technik ermöglicht auch mehre Speichervorgänge, bei denen aber nur Daten hinzugefügt werden. Die **CD-RW** kann

dagegen mehrfach beschrieben werden. Sie muss allerdings jedes Mal vorher gelöscht (formatiert) werden. Beiden gemeinsam ist die einzige spiralförmige Spur (wie bei der guten alten Schallplatte).

Die **DVD-ROM** ist die ursprüngliche *Read-Only*-DVD. DVDs können ein- oder zweilagig sowie ein- oder zweiseitig beschrieben sein. Daraus ergibt sich eine mögliche Kapazität zwischen rund 4,5 und 18 GByte. Die meisten handelsüblichen Video-DVDs sind zweilagig, aber einseitig beschrieben. Es gibt zwei unterschiedliche Verfahren, DVDs einfach (R) oder mehrfach (RW) zu beschreiben: **DVD-R, DVD-RW, DVD+R** und **DVD+RW**. Das DVD-R[W]-Format ist dem CD-R[W]-Format sehr ähnlich und stellt eine größtmögliche Kompatibilität mit DVD-Playern sicher.

Das DVD+R[W]-Format erlaubt von der Spezifikation her einen freien Zugriff auf alle Datensektoren (ähnlich wie DVD-RAM). Allerdings entsprechen noch nicht alle Laufwerke dieser Spezifikation. Außerdem nutzen nicht alle Brennprogramme diese Möglichkeiten. DVD+RWs müssen nur einmal formatiert werden (und nicht vor jedem neuerlichen Beschreiben).

Die Verfahren unterscheiden sich auch durch unterschiedliche Kopierschutzsysteme für Videos. Bei DVD+R[W]s kümmert sich das Laufwerk um den Kopierschutz. Bei DVD-R[W]s basiert der Kopierschutz dagegen auf einer eindeutigen Seriennummer, die jeder Rohling aufweisen muss. Wenn Sie DVDs als Speichermedium für Ihre eigenen Daten verwenden, spielt der Kopierschutz (dezeit noch) keine Rolle. Bis Anfang 2004 konnte sich keines dieser Verfahren durchsetzen. Stattdessen werden DVD-Brenner immer populärer, die alle Verfahren beherrschen. Momentan können die meisten DVD-Brenner nur einlagige, einseitige DVDs beschreiben, zweilagige Brenner sind aber auch schon auf dem Markt.

Die **DVD-RAM** bietet eine interessante Alternative zu DVD-RWs bzw. DVD+RWs. Sie wurden speziell im Hinblick auf die Speicherung von Daten optimiert. Anders als bei den RW-Formaten kann jeder Sektor der DVD jederzeit geändert werden. Ein weiterer Vorteil besteht darin, dass die Spezifikation bis zu eine Million Schreibzyklen vorsieht (während bei den RW-Verfahren theoretisch nach 1000 Zyklen, in der Praxis aber oft schon viel früher Schluss ist). Während zum Schreiben von DVD±R[W]s spezielle Software erforderlich ist, werden DVD-RAMs wie Festplatten behandelt: Sie können also mit `mke2fs` ein Linux-Dateisystem anlegen, die DVD mit `mount` in das Dateisystem einbinden und dann wie bei einer Festplatte Dateien und Verzeichnisse nach Belieben erzeugen und wieder löschen.

Der neue Standard **Mount Rainier (MRW)** (Mount Rainier ist ein Vulkan in der Nähe von Seattle und Redmont) soll in naher Zukunft dafür sorgen, dass DVD-RWs und DVD+RWs ähnlich komfortabel wie DVD-RAMs zu nutzen sind.

CD-Rs, DVD-Rs und DVD+Rs können normalerweise nur ein einziges Mal beschrieben werden. Die **Multi-Session-Technik** erlaubt das Aufteilen des Brennvorgangs in mehrere Teile. So ist es möglich, in der ersten Session einige Dateien zu speichern und in den weiteren Sessions neue Dateien hinzu-

zufügen. Bereits gespeicherte Daten können aber nicht verändert werden. Erst wenn die CD oder DVD explizit abgeschlossen wird (*close session*), sind keine weiteren Sessions mehr möglich. Leider bereiten Multi-Session-CDs und in noch viel stärkerem Ausmaß Multi-Session-DVDs oft Probleme beim Auslesen: Viele CD-ROM-Geräte können nur abgeschlossene Multi-Session-CDs lesen. Noch schlechter sieht es bei DVD-ROM-Laufwerken aus, die mit Multi-Session-DVDs oft gar nichts anfangen oder nur die erste Session lesen können.

Es hat sich eingebürgert, die Geschwindigkeit von CD- und DVD-Laufwerken mit einem Faktor anzugeben, der angibt, wie viel mal schneller Daten gelesen bzw. geschrieben werden können als beim gewöhnlichen Abspielen von Audio-CDs oder Video-DVDs (CD: 150 KByte/s, DVD: 1385 KByte/s).

Daten-CDs bzw. -DVDs enthalten meist ein Dateisystem nach ISO-9660. Wird der gesamte Inhalt dieses Dateisystem in einer einzigen Datei gespeichert, liegt eine so genannte ISO-Datei (ein **ISO-Image**) vor. Fast alle Brennprogramme bieten die Möglichkeit, eine derartige Datei direkt zu brennen. ISO-Images sind eine beliebte Möglichkeit, den Inhalt von CDs oder DVDs im Internet zur Verfügung zu stellen. Wenn die Anwenderin eine entsprechende CD benötigt, lädt sie einfach die ISO-Datei herunter und brennt die CD selbst.

7.6.1 CD- und DVD-Laufwerke korrekt konfigurieren

Beginnend mit Kernel 2.6 ist für das Brennen von CDs und DVDs keine SCSI-Emulation mehr erforderlich (wie bei den vorhergehenden Kernel-Versionen 2.0 bis 2.4). Allerdings müssen die Brennprogramme an den Kernel 2.6 angepasst werden, d. h. Sie müssen nach einem Kernel-Update eventuell auch neue Versionen von `cdrecord` etc. installieren.

Wenn es sich bei Ihren CD- bzw. DVD-Laufwerken um SCSI-, USB- oder Firewire-Geräte handelt, klappt normalerweise alles auf Anhieb. Alle Programme zum Brennen von CDs oder DVDs unter Linux unterstützen SCSI. USB- und Firewire-Geräte werden intern ebenfalls als SCSI-Geräte betrachtet. Linux kommt natürlich auch mit IDE-Laufwerken problemlos zurecht, allerdings müssen die Geräte bis einschließlich Kernel 2.4.*n* über eine SCSI-Emulationsschicht angesprochen werden. Das erfordert eine entsprechende Konfiguration, die bei den gängigen Distributionen automatisch erfolgen sollte. In der Praxis kommt es manchmal zu Problemen. Wenn sich das Brennprogramm darüber beschwert, dass es den CD- oder DVD-Brenner nicht findet, ist ein wenig Handarbeit angesagt:

- Je nachdem, ob Sie LILO oder GRUB als Boot-Loader verwenden, müssen Sie in `/etc/lilo.conf` (*append*-Zeile) bzw. in `/boot/grub/menu.lst` (*kernel*-Zeile) für jedes DVD-/CD-Laufwerk den Eintrag *hdx=ide-scsi* hinzufügen. Statt *hdx* geben Sie *hdb*, *hdc* oder *hdd* an, je nachdem, wie Ihr Laufwerk angeschlossen ist. (Mit `dmesg |grep hd` läßt sich der richtige Device-Name feststellen!) Bei LILO müssen Sie anschließend das Kommando `lilo`

ausführen. Die Einstellungen werden nach dem nächsten Rechnerstart wirksam.

- Da die CD- und DVD-Laufwerke nun als SCSI-Laufwerke betrachtet werden, ändert sich deren Device-Name von `/dev/hdx` in `/dev/scd0` (erstes CD/DVD-Laufwerk) und `/dev/scd1` (zweites CD/DVD-Laufwerk). Alternativ können die Laufwerke auch unter den Namen `/dev/sr0`, `/dev/sr1` etc. angesprochen werden.

  Je nach Konfiguration müssen Sie entweder direkt die Datei `/etc/fstab` ändern oder die Links `/dev/cdrom*` bzw. `/dev/dvd*` anpassen. Bei Suse lassen sich die Laufwerke mit YaST2 neu einrichten (Modul *Hardware* → *CD-Laufwerke*).

Bei vielen Programmen wird das Device des CD- oder DVD-Brenners in der Form `/dev/scd0` oder `/dev/sr0` angegeben, wie dies unter Linux üblich ist. Es gibt aber auch einige Kommandos (z. B. `cdrecord` und `cdrdao`), bei denen das Device durch drei Zahlen angegeben wird. Diese Zahlen bezeichnen den SCSI-Bus (meist 0), das SCSI-Gerät und die *logical unit number* (kurz LUN, meist ebenfalls 0). Diese Device-Nummern können Sie am einfachsten mit `cat /proc/scsi/scsi` ermitteln:

```
root#  cat /proc/scsi/scsi
Attached devices:
Host: scsi0 Channel: 00 Id: 04 Lun: 00
  Vendor: TOSHIBA  Model: CD-ROM XM-3601TA Rev: 0725
  Type:   CD-ROM                           ANSI SCSI revision: 02
Host: scsi0 Channel: 00 Id: 05 Lun: 00
  Vendor: TEAC     Model: CD-R50S-000      Rev: 1.0E
  Type:   CD-ROM                           ANSI SCSI revision: 02
```

Im obigen Beispiel gibt es ein CD-ROM-Laufwerk (Toshiba) und ein CD-R-Laufwerk (Teac). Die korrekte Bezeichnung für das Teac-CD-R-Laufwerk lautet `dev=0,5,0`. Wenn `cdrecord` installiert ist, können die Device-Nummern auch durch `cdrecord -scanbus` ermittelt werden.

Auf moderner Hardware sollte das Brennen von CDs und DVDs normalerweise ohne Probleme funktionieren. Sollte das nicht der Fall sein, beachten Sie folgende Tipps:

- Führen Sie während des Brennvorgangs keine Programme aus, die viel CPU-Leistung benötigen oder die häufig auf die Festplatte zugreifen.
- Aktivieren Sie unbedingt den DMA-Modus für alle Festplatten, falls dies nicht per Default der Fall ist.
- Experimentieren Sie auch mit dem DMA-Modus für die CD- bzw. DVD-Geräte. Hier gibt es leider keine generelle Empfehlung, ob es besser ist, diesen Modus zu aktivieren oder nicht. Der DMA-Modus bewirkt eine schnellere

und vor allem für die CPU weniger aufwändige Übertragung der Daten zum Laufwerk. Allerdings kann der DMA-Modus bei manchen Laufwerken auch Probleme verursachen (hardware-, distributions- und kernelabhängig).

- Brennen Sie die CDs/DVDs mit einer kleineren Brenngeschwindigkeit.
- Vermeiden Sie das Brennen *on the fly*.
- Verlassen Sie sich nicht unbedingt auf die Burnfree-Funktion, die selbst dann korrekte CDs bzw. DVDs garantieren sollte, wenn während des Brennens der interne Puffer des Brennlaufwerks leer wird.

7.7 Benutzeroberflächen

Unter Unix/Linux ist es üblich, für kleine, überschaubare Aufgaben jeweils einzelne Werkzeuge zu schaffen. Deshalb gibt es eine Fülle von kommandoorientierten Werkzeugen, beispielsweise `mkisofs`, `cdrecord`, `cdrdao`, `growisofs` etc. Auf diese kommen wir weiter unten zu sprechen.

Wenn Sie rasch eine CD kopieren oder ein Backup Ihrer Daten auf eine DVD brennen möchten, haben Sie vermutlich keine Lust, sich mit den Details und den unzähligen Optionen all dieser Kommandos auseinander zu setzen. Deshalb gibt es Benutzeroberflächen, die einfach zu bedienen sind, intern aber auf Kommandos wie `cdrecord` zurückgreifen.

Dieser Abschnitt stellt die beiden wichtigsten Benutzeroberflächen vor, `k3b` (KDE) und Nautilus (Gnome). Wenn Sie damit nicht glücklich werden, können Sie beispielsweise auch den Klassiker `xcdroast`, `CDBO` (CD Bake Oven) oder `gtoaster` ausprobieren.

7.7.1 K3b (KDE)

`k3b` hat sich in relativ kurzer Zeit zum leistungsfähigsten Brennprogramm entwickelt, das unter Linux verfügbar ist. Sie können damit CDs und alle Arten von DVDs brennen. Die Durchführung von Standardaufgaben (CD kopieren, Daten-CD oder Daten-DVD erzeugen etc.) ist extrem einfach (*http://www.k3b.org/*).

Mit *Einstellungen* → *K3b einrichten* gelangen Sie in den Konfigurationsdialog. Alle weiteren Tipps beziehen sich auf diesen Dialog.

- **Hardware-Erkennung:** `k3b` versucht, alle CD- und DVD-Laufwerke automatisch zu erkennen. Die Hardware-Suche können Sie im Dialogblatt *Geräte* wiederholen (Button *Aktualisieren*).
- **Software-Erkennung:** `k3b` greift auf unzählige externe Tools zurück. Wenn Sie ein derartiges Programm nachträglich installieren, wiederholen Sie die

Software-Suche im Dialogblatt *Programme → Suchpfad*, indem Sie den Button *Suchen* anklicken. Sie können in diesem Dialog auch die Suchpfade angeben, in denen `k3b` nach den Programmen sucht.

- **cdrdao-Option:** `k3b` greift unter anderem auf das Kommando `cdrdao` zurück. Dieses Kommando liefert bei vielen DVD- und CD-Geräten die Fehlermeldung *kein cdrdao-Treiber gefunden*. Um das zu vermeiden, geben Sie in *Programme → Benutzerdefinierte Parameter* beim Programm `cdrdao` den Parameter `--driver generic-mmc` an. (Für die meisten Geräte, die `cdrdao` nicht erkennt, ist `generic-mmc` oder `generic-mmc-raw` die richtige Einstellung.
- **CDDB:** Falls CDDB nicht funktioniert (also der Zugriff auf eine CD-Titeldatenbank im Internet, müssen Sie die Funktion im Dialogblatt *CDDB → Entfernt* konfigurieren. Dazu löschen Sie eventuell vorhandene Einträge und definieren dann einen neuen, wobei Sie als Server-Typ *Cddbp* angeben, als Adresse *freedb.freedb.org* und als Port-Nummer 8880. Achten Sie darauf, dass die Option *Entfernte CDDB-Anfragen aktivieren* gesetzt ist.
- **Projekt- und Aktionseinstellungen:** Zu allen Projekt- und Aktionstypen (z. B. CD kopieren) gibt es einen eigenen Dialog mit diversen Einstellungen für die Brenngeschwindigkeit, für den Ort temporärer Dateien etc. Damit Sie die Einstellungen nicht immer wieder neu eingeben müssen, können Sie die durchgeführten Einstellungen speichern (Button *Benutzervoreinstellungen speichern*).
- **Benachrichtigung durch Klänge:** Wenn Sie keine Lust auf plötzliche Fanfarenklänge haben, sollten Sie im Dialogblatt *Benachrichtigungen* den Button *Alle deaktivieren* anklicken.

Eventuell meldet `k3b` beim Start, dass die Programme `cdrecord`, `cdread` und `cdrdao` keine Setuid-Privilegien haben. Das hat zwei Konsequenzen: Die Programme funktionieren nur dann, wenn die aktuelle Benutzerin Schreib- und Leserechte für die CD- bzw. DVD-Devices hat. Die Programme können kein *real time sheduling* verwenden, um damit eine höhere Priorität bei der Prozessausführung zu erlangen. Im Regelfall können Sie die Warnung einfach ignorieren. Die Alternative besteht darin, dass Sie als `root` für die betroffenen Programme `chmod u+s Programmname` ausführen – Sie handeln sich damit aber ein Sicherheitsrisiko ein.

Suse liefert eine speziell kompilierte Version von `k3b` aus, die keine Setuid-Warnungen wie bei der Default-Version liefert. Hier kümmert sich das Programm `resmgr` darum, dass die CD- und DVD-Devices die richtigen Zugriffsrechte haben.

Der KDE-Brenner `k3b` kann auf zwei Arten eingesetzt werden: Einerseits gibt es eine Reihe von Projekttypen zur Durchführung verschiedener Aufgaben, z. B. zum Erstellen einer Audio-CD oder einer Daten-DVD. Derartige Projekte werden mit *Datei → Neues Projekt* gestartet. Anschließend fügen Sie per

Drag&Drop die zu speichernden Daten in das Projektfenster ein. Der Button *Brennen* führt dann zu einem Dialog, in dem Sie alle weiteren Einstellungen durchführen können.

Andererseits können eine Reihe einfacherer Aufgaben, z. B. das Kopieren einer CD oder das Löschen eines RW-Mediums, über das *Extras*-Menü durchgeführt werden. Für diese Aufgaben gibt es kein Projektfenster, alle Einstellungen werden direkt in einem Dialog durchgeführt. Besonders oft benötigte Standardaufgaben können auch direkt durch das Anklicken von Icons in der `k3b`-Symbolleiste oder im Startfenster ausgelöst werden.

`k3b` bietet gleich drei verschiedene Möglichkeiten, eine CD zu kopieren:

- *Extras* → *CD* → *Kopieren* verwendet zum Kopieren das Programm `cdrdao`. In den meisten Fällen ist das der beste und schnellste Weg zu einer Kopie.
- *Extras* → *CD* → *Klonen* liest die Daten mit `readcd` und schreibt sie mit `cdrecord` (etwas langsamer).
- Die dritte Variante besteht darin, zuerst einzelne Stücke der Audio-CD einzulesen. Dazu klicken Sie das CD-Icon an, wählen die gewünschten Tracks aus und klicken dann den Button *Auslesen starten* an. `k3b` liest die Tracks selbst ein und speichert sie als Dateien (per Default im Format WAV) in einem Verzeichnis der Festplatte. Diese Dateien können Sie dann im nächsten Schritt auf eine CD brennen (siehe unten). Die Vorgehensweise ist also etwas umständlicher, bietet aber den Vorteil, dass Sie selbst die zu kopierenden Tracks auswählen können.

Um eine **Audio-CD** aus WAV-, MP3- oder Ogg-Vorbis-Dateien zu erzeugen, starten Sie ein neues *Audio-CD-Projekt* und verschieben die Audio-Dateien von der Festplatte in das Projektfenster. MP3- bzw. Ogg-Vorbis-Dateien werden vor dem Brennen automatisch in das WAV-Format umgewandelt, so dass die resultierende CD auf jedem CD-Player abgespielt werden kann. (Der Player braucht also keine MP3-Funktionen. Eine so genannte MP3-CD ist eine Daten-CD, die MP3-Dateien enthält.)

Zum Erstellen einer **Daten-CD** bzw. **Daten-DVD** starten Sie ein neues *Daten-CD-* oder *-DVD*-Projekt und verschieben die zu speichernden Dateien und Verzeichnisse in das Projektfenster. Vorsicht ist beim Umgang mit symbolischen Links geboten: Per Default speichert `k3b` auf der CD nur die Informationen über den Link an sich, nicht aber die Daten, auf die der Link verweist. Zeigt ein Link auf ein anderes Verzeichnis, ist dessen Inhalt nicht auf der CD (Abhilfe siehe unten).

Im Brenndialog haben Sie anschließend sehr viele Einstellmöglichkeiten, von denen hier nur die wichtigsten genannt werden:

- *Einstellungen* → *Mehrfachsitzung*: Hier können Sie neue Multi-Session-CDs bzw. -DVDs erzeugen bzw. erweitern.

- *Dateisystem*: Hier geben Sie an, ob die CD oder DVD die Unix/Linux-kompatible Rockridge-Extension (Default-Einstellung) oder die Windows-kompatible Joliet-Extension nutzen soll. Diese Erweiterungen zum ursprünglichen CD-Standard ermöglichen die Speicherung langer Dateinamen. Sie können auch beide Erweiterungen gleichzeitig aktivieren, um eine größtmögliche Austauschbarkeit der CD zu erreichen.
- *Dateisystem → Symbolische Verknüpfungen*: Hier können Sie angeben, ob symbolische Links gespeichert werden oder nicht.
- *Fortgeschritten → Symbolische Verknüpfungen folgen*: Die Option bewirkt, dass nicht symbolische Links an sich gespeichert werden, sondern die Daten, auf die die Links verweisen.
- Weitere Brennoptionen finden Sie im zentralen Konfigurationsdialog *Einstellungen → K3b einrichten* im Dialogblatt *Brennen* (z. B. automatisches Löschen von RW-Datenträgern oder Einstellend der Zwischenpuffergröße von `cdrecord`).

Überprüfen Sie nach dem Brennen, ob die Daten wirklich korrekt gespeichert worden sind! (Nicht jeder DVD-Rohling verträgt sich mit jedem Laufwerk.) Am besten aktivieren Sie dazu im Dialogblatt *Brennen* die Option *Geschriebene Daten überprüfen*. Nachträglich können Sie mit dem Linux-Kommando `diff` den Inhalt zweier Verzeichnisse (z. B. auf der DVD und auf der Festplatte) vergleichen.

7.7.2 Nautilus (Gnome)

Der Gnome-Dateimanager kann durch das Zusatzmodul `nautilus-cd-burner` um eine einfache Funktion zum Brennen von Daten-CDs erweitert werden. Die Bedienung ist denkbar einfach: Sie geben den Pfad `burn:///` ein bzw. führen *Gehe zu → CD-Ersteller* aus. Anschließend verschieben Sie per Drag&Drop alle zu speichernden Dateien bzw. Verzeichnisse von einem anderen Nautilus-Fenster in das Fenster `burn:///`.

Mit dem Button *Inhalt auf CD schreiben* gelangen Sie in den Brenndialog. Dort können Sie die gewünschte Geschwindigkeit angeben. Mit der Option *CD-Inhalt löschen* werden CD-RWs vor dem Schreiben gelöscht. Die Option *Diese Dateien erneut für eine andere CD verwenden* ermöglicht es, mehrere CDs hintereinander zu brennen, ohne die Dateien und Verzeichnisse jedes Mal neu in das `burn:///`-Fenster zu verschieben. (Wenn diese Option nicht gesetzt wird, löscht Nautilus den Inhalt des `burn:///`-Fensters nach dem Brennen.) Intern verwendet Nautilus `cdrecord` zum Brennen der CD.

Die Nautilus-Brennfunktionen sind zwar einfach zu bedienen, aber im Vergleich zu `k3b` auch sehr eingeschränkt.

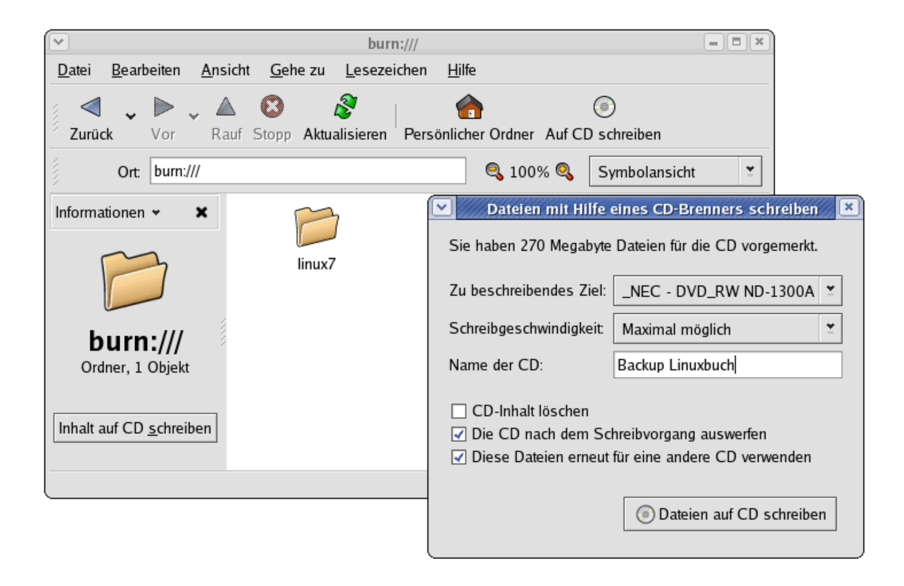


Abbildung 7.2: CDs mit Nautilus brennen

7.7.3 Brenner-Tools

Dieser Abschnitt stellt die wichtigsten Kommandos zum Erstellen eines ISO-Images sowie zum Schreiben und Lesen von CDs, Audio-CDs und DVDs vor. Es sind dies:

Tools zum Lesen und Schreiben von CDs und DVDs	
`mkisofs`	Dateisystem für CD/DVD erzeugen
`cdrecord`	Daten-CD schreiben, CD-RW löschen
`readcd`	Daten-CD auslesen
`cdrdao`	Audio-CD lesen und schreiben (DAO-Modus)
`cdparanoia`	Audio-CD lesen (Audio-Tracks lesen)
`cdda2wav`	Audio-CD lesen (Audio-Tracks lesen)
`dvd+rw-format`	DVD+RW und DVD-RW löschen
`growisofs`	DVD schreiben

Die Programme `cdparanoia` und `cdda2wav` wurden bereits auf Seite 298 kurz vorgestellt. Werkzeuge zum Lesen und Neukodieren von Video-DVDs (etwa zur anschließenden Speicherung auf einer CD) werden in diesem Buch nicht behandelt. Sie finden derartige Tools aber im Internet unter dem Schlagwort *dvd rip*.

mkisofs

`mkisofs` erzeugt eine Datei (ein so genanntes ISO-Image), die ein vollständiges ISO-9660-Dateisystem enthält. Diese Datei kann anschließend auf eine CD oder DVD gebrannt werden. ISO-Dateien sind somit die Grundlage aller Daten-CDs und -DVDs.

mkisofs `[optionen] verzeichnis1 verzeichnis2`

Grundsätzlich werden an `mkisofs` alle Verzeichnisse übergeben, deren Inhalt in das Wurzelverzeichnis des ISO-9660-Dateisystems eingefügt werden soll. Die Verzeichnisse selbst werden nicht Teil des ISO-Dateisystems! (Wenn Sie das möchten, müssen Sie die Option *-graft-points* verwenden und die Verzeichnisse in der Form `verz1=verz1 verz2=verz2` angeben.)

`mkisofs` schreibt die resultierende ISO-Datei in die Standardausgabe. Oft wird die Ausgabe mit `-o dateiname` in eine Datei umgeleitet. Daneben gibt es zahllose weitere Optionen, von denen in den Tabellen 7.1 und 7.2 nur die wichtigsten erwähnt sind.

ISO-9660 verwendet einen eigenen, sehr limitierten Zeichensatz, der nur wenige Nicht-ASCII-Zeichen zulässt. Erst die Rockridge-Erweiterungen erlauben beinahe jede Zeichenkette als Dateiname. Allerdings enthalten Rockridge-CDs keine Information darüber, in welchem Zeichensatz das ISO-Image erstellt wurde. Normalerweise lässt `mkisofs` die Zeichensatzkodierung unverändert, wenn die Rockridge-Extension verwendet wird. Für Joliet muss allerdings eine Umkodierung erfolgen, weil der Zeichensatz für Joliet mit Unicode/UTF16 fest eingestellt ist. Mit `-input-charset` können Sie angeben, welchen Zeichensatz die lokalen Dateinamen verwenden (Default: Latin-1). `mkisofs -input-charset help` liefert eine Liste der möglichen Zeichensätze.

Aus diesem Grund ist es mit `mkisofs` unter einer UTF8-Linux-Distribution (Red Hat) unmöglich, eine Joliet-CD zu erzeugen, die korrekte Dateinamen mit internationalen Sonderzeichen enthält. Mit einem Zeichensatz erzeugte Rockridge-CDs machen beim Lesen mit einem anderen Zeichensatz Probleme. Wenn Ihre Dateinamen also internationale Sonderzeichen enthalten, müssen Sie bei der Verwendung der CD auf einem anderen Rechner mit Problemen rechnen. Falls Ihre Distribution einen 8-Bit-Zeichensatz ungleich Latin-1 verwendet, können Sie die Probleme eventuell umgehen, indem Sie den aktiven Zeichensatz mit `-input-charset` und den gewünschten Zielzeichensatz auf der CD mit `-output-charset` angeben. Die Umkodierung ist aber insofern beschränkt, als `mkisofs` (noch) nicht mit Unicode-Dateinamen zurechtkommt.

CDs, die eine größtmögliche Kompatibilität mit Apple-Rechnern aufweisen sollen, müssen das HFS-Dateisystem verwenden. Je nachdem, welche Optionen Sie an `mkisofs` übergeben, können Sie derartige CDs auf zwei Arten erzeugen: `mkisofs -hfs ...` erzeugt eine Hybrid-CD, die sowohl das ISO-9660- als auch das HFS-Dateisystem verwendet. Die eigentlichen Dateien werden aber nur

Tabelle 7.1: Optionen von mkisofs, Teil 1

-b *datei*	Die angegebene Datei wird als Boot-Image gespeichert. Beim Rechnerstart wird diese Datei automatisch ausgeführt (sofern das BIOS entsprechend konfiguriert ist). Wenn nicht außerdem `-hard-disk-boot` angegeben wird, erwartet `mkisofs` ein Disketten-Image, dessen Größe exakt einer 3.5- oder 5.25-Zoll-Diskette entspricht (1,2, 1,44 oder 2,88 MByte).
-c *name*	Gibt an, unter welchem Dateinamen der so genannte Boot-Katalog auf der CD gespeichert werden soll. Der Katalog wird von `mkisofs` erzeugt. Sie müssen bei der Angabe des Namens lediglich darauf achten, dass es keinen Konflikt mit einer vorhandenen Datei gibt (z. B. `boot.catalog`).
-D	Der ISO-Standard erlaubt maximal acht Verzeichnisebenen. `mkisofs` hält sich normalerweise an diesen Standard und verändert gegebenenfalls die Verzeichnishierarchie (und die Namen der betroffenen Dateien). Die Option `-D` verhindert das, was aber auf manchen Systemen, die eine exakte Einhaltung des ISO-Standards verlangen, zu Problemen führt.
-dvd-video	Das ISO-Image berücksichtigt die Regeln für Video-DVDs (UDF-Unterstützung, korrekte Sortierung der Dateien etc.). Achten Sie darauf, dass die Dateinamen der Video-Dateien in Großbuchstaben angegeben werden müssen!
-f	Symbolische Links werden verfolgt, d. h. der Inhalt der so angegebenen Dateien oder Unterverzeichnisse wird in das ISO-Image mit aufgenommen. (Ohne diese Option wird nur der Link an sich im ISO-Image gespeichert. Symbolische Links können allerdings nur auf CDs mit Rockridge-Extension abgebildet werden.)
-graft-points	Verzeichnisangaben können nun in der Art `iso=real` erfolgen. Das bedeutet, dass das Verzeichnis `real` im ISO-Image den Namen `iso` bekommt. Sie können mit diesem Mechanismus auch den Pfad von Verzeichnissen ändern oder einzelne Dateien umbenennen. Oft wird die Option nur deswegen verwendet, um zu erreichen, dass das zu speichernde Verzeichnis an sich (und nicht nur sein Inhalt) in das ISO-Image aufgenommen wird. In diesem Fall stimmen der ISO-Name und der reale Name überein (`verz1=verz1 verz2=verz2`).

einmal gespeichert. `mkisofs -apple ...` erzeugt eine ISO-9660-CD mit Apple-Extension. Durch diverse HFS-spezifische Optionen können Sie weitere Details einstellen (siehe `man mkisofs`).

Tabelle 7.2: Optionen von mkisofs, Teil 2

-hard-disk-boot	Das mit `-b` angegebene Boot-Image ist ein Festplatten-Image. Das Image muss mit einem MBR (*master boot record*) beginnen und eine Partition enthalten.
-J	Das ISO-Image nutzt die Joliet-Extension (Microsoft), um Informationen über lange Dateinamen zu speichern. Dateinamen werden in Unicode (UTF-16) kodiert. Die Option kann auch in Kombination mit `-R` verwendet werden – dann ist das Ergebnis eine Hybrid-CD, bei der lange Dateinamen sowohl unter Unix als auch unter Windows sichtbar sind.
-joliet-long	Die Joliet-Extension beschränkt Dateinamen normalerweise auf 64 Zeichen. Mit dieser Option sind bis zu 103 Zeichen erlaubt.
-nobak	Die Option bewirkt, dass alle Dateien ignoriert werden , die die Zeichen `#` oder `~` enthalten oder mit `.bak` enden.
-o *name*	Das ISO-Image wird in Datei *name* gespeichert.
-R	Das ISO-Image nutzt die Rockridge-Extension (Unix), um Informationen über lange Dateinamen zu speichern. Die Option kann auch in Kombination mit `-J` verwendet werden – dann ist das Ergebnis eine Hybrid-CD.
-r	Wie `-R`, es werden aber für alle Dateien UID und GID auf 0 gesetzt (entspricht `root`). Gleichzeitig werden alle R-Zugriffsbits gesetzt, außerdem alle X-Zugriffsbits, wenn in der ursprünglichen Datei zumindest ein X-Bit gesetzt war. Wenn Sie die Zugriffsrechte exakter steuern möchten, können Sie dies mit den Optionen `-uid`, `-gid`, `-dir-mode`, `-file-mode` und `-new-file-mode` tun (siehe `man mkisofs`).
-udf	Diese Option ergänzt das ISO-Image um UDF-Verwaltungsinformationen. Wenn Sie Video-DVDs erzeugen möchten, verwenden Sie statt `-udf` besser die Option `-dvd-video`.
-V *name*	Diese Option gibt die Volume-ID an, also den Namen der CD-ROM (maximal 32 Zeichen), wie er beispielsweise unter Windows angezeigt wird.

Beispiele: Das folgende Kommando schreibt alle Dateien innerhalb des `/master`-Verzeichnisses in die Datei `/data/iso.img`. Das Verzeichnis `master` ist selbst *kein* Verzeichnis im ISO-Image. Das ISO-Image bekommt den Namen `Linux`. (Wenn Sie aus dem ISO-Image eine CD brennen, gilt diese Zeichenkette als CD-Name.)

```
root# mkisofs -o /data/iso.img -r -V Linux /master
```

Wie oben, allerdings wird diesmal eine bootfähige CD erstellt:

```
root# mkisofs -o /data/iso.img -r  /master \
    > -b /images/boot.img -c boot.catalog
```

Wie das erste Beispiel, allerdings ist das Verzeichnis `master` nun selbst ein Verzeichnis im ISO-Image:

```
root# mkisofs -o /data/iso.img -r -graft-points /master=/master
```

Mit dem so genannten *Loopback device* der Linux-Kernels können Sie eine Datei als Dateisystem betrachten und mit `mount` in den Verzeichnisbaum einbinden. Die Loopback-Funktion ist im Kernel-Modul `loop.o` versteckt, das bei allen gängigen Distributionen zur Verfügung steht. Falls das Modul nicht automatisch geladen wird, müssen Sie eventuell mit `modprobe` nachhelfen. Mit dem folgenden Kommando können Sie die Datei `iso.img` im Read-Only-Modus in das Dateisystem einbinden:

```
root# mkdir /iso-test
root# mount -t iso9660 -o loop,ro /data/iso.img /iso-test/
```

Über das Verzeichnis `iso-test` können Sie jetzt den Inhalt der zukünftigen CD-ROM überprüfen.

dd und cdread (Daten-CDs und -DVDs lesen)

Wenn Sie eine Daten-CD oder -DVD (keine Audio-CD!) unverändert kopieren möchten, reicht ein einziges Kommando aus, um die erforderliche ISO-Datei zu erzeugen:

```
user$ dd if=/dev/cdrom of=/usr/local/iso.img bs=2048
```

Eine Variante zu `dd` ist das Kommando `readcd`. Es verwendet SCSI-Kommandos zum Auslesen der CD und sollte zum gleichen Ergebnis kommen wie `dd`.

```
user$ readcd dev=0,0 f=iso.img
```

Der Vorteil von `readcd` besteht darin, dass es auch eine TOC-Datei erstellen kann (bei Audio-CDs mit der Option `-clone`) und dass das Programm je nach Optionen auf unterschiedliche Weise mit Lesefehlern umgehen kann (Optionen `-noerror` und `-noclone`, siehe die Manual-Seite). Mit `-w` kann `readcd` auch zum Schreiben von CDs verwendet werden.

cdrecord (Daten-CDs schreiben)

`cdrecord` überträgt ein ISO-Image (Daten-CDs) oder Audio-Dateien (für Audio-CDs) an das CD-R-Laufwerk. `cdrecord` ist prinzipiell multisession-fähig, d. h. die CD kann in mehreren Durchgängen erweitert werden.

```
cdrecord [optionen] dev=xxx isoimage
cdrecord [optionen] dev=xxx audiofiles
```

Das Verhalten von `cdrecord` wird durch zahlreiche Optionen gesteuert, von denen in den Tabellen 7.3 und 7.4 die wichtigsten vorgestellt werden. Beachten Sie, dass es Optionen mit und ohne vorangestellten Bindestrich gibt!

Tabelle 7.3: Optionen von cdrecord, Teil 1

blank=all—fast	Mit dieser Option löscht `cdrecord` das CD-RW-Medium.
dev=scsi,id,lun	Diese Option gibt das CD-R-Laufwerk an. Unter Kernel 2.4 muss eine Kombination aus drei Nummern angegeben werden (siehe auch Seite 318), und zwar die SCSI-Bus-Nummer (meistens 0), die SCSI-ID des Gerätes und schließlich die *logical unit number* (kurz LUN, ebenfalls meist 0).
dev=/dev/xxx	Ab Kernel 2.6 geben Sie einfach den Device-Namen des Laufwerks an, z. B. `/dev/hdc`. (Das setzt eine Kernel-2.6-kompatible Version von `cdrecord` voraus.)
driver=name	Diese Option gibt an, welcher Treiber zum Zugriff auf das CD-Laufwerk verwendet werden soll. Das ist nur selten notwendig, weil `cdrecord` den korrekten Treiber meist selbst erkennt. `cdrecord driver=help` liefert eine Liste der verfügbaren Treiber. Mit `cdrecord -checkdrive dev=`*n*, *n*, *n* können Sie versuchen, den Typ des vorhandenen CD-R-Laufwerks zu erkennen.
driveropts=burnfree	Die Option aktiviert einen Schutzmechanismus zur Vermeidung von Buffer-Underrun-Fehlern.
fs=n	Diese Option gibt die Größe des FIFO-Zwischenspeichers an, der zur Pufferung der Datenübertragung zwischen Festplatte und CD dient. Vernünftige Werte liegen zwischen zwei und acht MByte (also `fs=2m` oder `fs=8m`). Der Defaultwert beträgt vier MByte.
speed=n	Diese Option gibt die Schreibgeschwindigkeit an.

Daten-CD brennen: Mit den beiden folgenden Kommandos wird zuerst das Brennen einer Daten-CD simuliert (`-dummy`) und dann tatsächlich durchgeführt:

```
root# cdrecord -dummy -v speed=16 dev=0,5,0 iso.img
```

Tabelle 7.4: Optionen von cdrecord, Teil 2

Option	Beschreibung
-audio	Bei den angegebenen Dateinamen handelt es sich um Audio-Dateien (üblicherweise `*.au`- oder `*.wav`-Dateien).
-dummy	Der Schreibprozess wird nur simuliert. Ideal zum Testen!
-dao	Diese Option aktiviert den Disk-at-Once-Modus. Das verhindert Pausen zwischen den Tracks von Audio-CDs und auf diese Weise hergestellte Daten-CDs sind als Master für die Massenproduktion geeignet.
-eject	Wirft die CD nach dem Schreibprozess aus. Manche CD-R-Laufwerke verlangen den Auswurf, bevor die nächste CD geschrieben werden kann, selbst wenn das Schreiben zuletzt nur mit `-dummy` simuliert wurde.
-multi	Diese Option muss bei allen Sessions einer Multi-Session-CD mit Ausnahme der letzten Session angegeben werden.
-msinfo	Liest vorhandene Session-Informationen von der CD. Die Option muss ab der zweiten Session angegeben werden.
-pad	Diese Option bewirkt in Kombination mit `-audio`, dass die Länge von Audio-Dateien auf ein Vielfaches von 2352 Byte ergänzt wird (wenn das nicht ohnedies der Fall ist).
-scanbus	Mit dieser Option sucht `cdrecord` nach CD-R-Laufwerken am SCSI-Bus.
-swab	Vertauscht die Byte-Reihenfolge der Daten. Verschiedene CD-Rekorder erwarten eine unterschiedliche Reihenfolge der Bytes in Datenworten. `cdrecord` erkennt das selbstständig, so dass die Option im Regelfall nicht angegeben werden muss.
-v	`cdrecord` gibt ausführliche Meldungen aus, was gerade geschieht.

`root#` **cdrecord -v speed=16 dev=0,5,0 iso.img**

Auf schnellen Rechnern können Sie `mkisofs` und `cdrecord` mit einer Pipe verbinden. Dadurch sparen Sie den Platz für das ISO-Image:

`root#` **mkisofs -r /master | cdrecord -v speed=16 dev=0,5,0 -**

Audio-CD brennen: Das folgende Kommando erzeugt eine Audio-CD. Die Ausgangsdaten liegen als `*.wav`-Dateien vor. Die Dateien werden in alphabetischer Reihenfolge verarbeitet. Wenn Sie eine andere Reihenfolge wünschen, müssen Sie die Dateien der Reihe nach angeben.

`root#` **cdrecord -v speed=16 dev=0,5,0 -pad -dao -audio *.wav**

`cdrecord-ProDVD` ist eine kommerzielle, DVD-taugliche Version von `cdrecord` (DVD-R[W] und DVD+R[W]). Privatanwender dürfen diese Version kostenlos nutzen, müssen dafür aber zweimal pro Jahr einen neuen Lizenzschlüssel installieren. Dieser Abschnitt geht nicht auf `cdrecord-ProDVD` ein, sondern beschreibt nur die CD-R-Funktionen von `cdrecord`. Mehr Informationen zu `cdrecord-ProDVD` finden Sie unter *http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html*.

Eine Variante zu `cdrecord-ProDVD` ist das Kommando `dvdrecord` aus dem GPL-Projekt `dvdrtools`. Es ist ebenfalls aus `cdrecord` entstanden, ist ganz ähnlich zu bedienen und kann DVD-Rs und DVD-RWs schreiben (*http://www.nongnu.org/dvdrtools/*).

Daten-CD kontrollieren

Mit dem folgenden Kommando wird der Inhalt der CD-ROM mit dem des `master`-Verzeichnisses Datei für Datei und Byte für Byte verglichen:

`root#` **diff -qrd /master /mnt/cdrom/ | less**

Dabei sind Fehlermeldungen aufgrund von symbolischen Links, die auf der CD nicht mehr an den richtigen Ort verweisen, zu erwarten. Ein echtes Warnsignal ist es hingegen, wenn einzelne Dateien gar nicht gelesen werden können (I/O-Error) oder wenn der Inhalt von Dateien abweicht.

Wenn Sie einfach nur testen möchten, ob alle Datenblöcke der CD gelesen werden können (ganz egal, welchen Inhalt sie haben), führen Sie das folgende Kommando aus. Dieser Test ist beispielsweise dann sinnvoll, wenn Sie eine CD bekommen haben, von der Sie vermuten, dass sie defekt ist.

`root#` **dd if=/dev/cdrom of=/dev/null**

cdrdao (Audio-CDs lesen und schreiben)

Das Programm `cdrdao` ist eine Alternative zu `cdrecord`. `cdrdao` ist zwar nicht so vielseitig, bietet dafür aber wesentlich mehr Optionen zum Lesen und Schreiben von Audio-CDs. Seinen Namen verdankt `cdrdao` dem Schreibmodus *Disk at Once* (kurz DAO).

Beim Schreiben von Audio-CDs setzt das Programm voraus, dass ein so genanntes TOC-File existiert (eine Art Inhaltsangabe für die CD, die angibt, welche Daten in welchem Format wie gespeichert werden). Ein derartiges TOC-File kann am einfachsten aus einer vorhandenen Audio-CD ermittelt werden (wenn diese kopiert werden soll).

`cdrdao` unterstützt verschiedene Treiber zur Kommunikation mit dem Laufwerk. Bei neuen Laufwerken erkennt `cdrdao` allerdings nicht, welchen Treiber es verwenden muss. In diesem Fall müssen Sie den Treiber mit

`--driver name` angeben. Bei fast allen modernen Laufwerken lautet der richtige Treiber `generic-mmc` oder `generic-mmc-raw`. Am besten fügen Sie in `/usr/share/cdrdao/drivers` eine neue Zeile ein, die aus vier Informationen besteht: dem Laufwerkstyp (R für *read only*, W für *write*), dem Herstellernamen, dem Gerätenamen und dem erforderlichen `cdrdao`-Treiber, z. B.

```
W|_NEC|DVD_RW ND-1300A|generic-mmc
```

In der Praxis besteht die gebräuchlichste Anwendung von `cdrdao` darin, Audio-CDs zu kopieren. Das erste `cdrdao`-Kommando erzeugt die Dateien `data.bin` (Inhalt der CD) und `data.toc` (Inhaltsverzeichnis). Das zweite Kommando schreibt diese Daten auf eine CD:

```
user$  cd tmp/
user$  cdrdao read-cd --device 0,0,0 data.toc
user$  cdrdao write --device 0,0,0 --buffers 64 data.toc
```

Zwei CD-Laufwerke und einen schnellen Rechner vorausgesetzt, können Sie die Kopie sogar in einem einzigen Schritt durchführen:

```
root#  cdrdao copy --source-device 0,0,0 --device 0,1,0 --buffers 64
```

Weitere Informationen zu `cdrdao` finden Sie in der Manual-Seite, in den sehr ausführlichen Dokumentationsdateien sowie auf der Website *http://cdrdao.sourceforge.net*.

dvd+rw-format (DVD-RWs und DVD+RWs formatieren)

`dvd+rw-format` ist ein Teil des Pakets `dvd+rw-tools`, das auch das Kommando `growisofs` enthält.

dvd+rw-format `[optionen] device`

`dvd+rw-format` formatiert DVD+RW- und DVD-RW-Medien. Als einziger Parameter wird normalerweise nur der Device-Name angegeben. Optional kann durch `-force` die Formatierung eines bereits formatierten Mediums erzwungen werden.

```
user$  dvd+rw-format /dev/sr0
```

Tipp

Anders als CD-RWs müssen DVD+RWs und DVD-RWs (im Modus *restricted overwrite*, siehe unten) nur vor der ersten Verwendung formatiert werden! Anschließend kann der Datenträger immer wieder neu durch *growisofs -Z device* beschrieben werden. Vorhandene Daten werden automatisch überschrieben. Wenn Sie `dvd+rw-format` dennoch auszuführen versuchen, zeigt das Programm eine entsprechende Warnung an.
Durch `-force` können Sie in jedem Fall eine Formatierung erzwingen. Diese Formatierung löscht allerdings die Daten nicht physikalisch. Falls Sie dies aus Datenschutzgründen wünschen, führen Sie besser `growisofs -Z device=/dev/null` aus.

DVD+RWs: `dvd+rw-format` formatiert den Rohling nur im Anfangsbereich. Wie weit der reicht, hängt vom Brenner ab. Wenn der Formatiervorgang also bei 11,5 Prozent (oder irgendeiner anderen Prozentzahl kleiner 100) endet, ist dies kein Fehler! Die Formatierung über den Anfangsbereich hinaus erfolgt automatisch durch das Laufwerk, sobald die DVD über den vorformatierten Bereich hinaus beschrieben wird.

DVD-RWs: Bei DVD-R[W]s gibt es drei unterschiedliche Brennmodi:

- *disk at once*: Dieser Modus bietet die größte Kompatibilität zu DVD-ROM-Geräten und sollte insbesondere für Videos verwendet werden. Der Modus wird von den *dvd+rw-tools* allerdings nicht unterstützt.
- *incremental sequential*: Dieser Modus eignet sich ebenfalls für Video-DVDs. Außerdem ermöglicht er es, Multi-Sessions ähnlich wie bei CD-Rs zu nutzen. Allerdings können viele DVD-ROM-Geräte nur die erste Session lesen.
- *restricted overwrite* (nur für DVD-RWs, nicht für DVD-Rs): Dieser Modus ermöglicht es, bereits beschriebene Bereiche der DVD neu zu beschreiben (*overwrite*). Es ist daher nicht erforderlich, DVD-RWs vor jedem Schreiben neu zu formatieren!

  Weiterhin können DVD-RWs in mehreren Schritten beschrieben werden. Aus Anwendersicht entspricht das einer Multi-Session-DVD. Für DVD-Lesegeräte sieht die DVD dennoch wie eine Single-Session-DVD aus. Daher gibt es im Regelfall weniger Probleme, wenn die DVD in einem anderen Laufwerk gelesen werden soll.

Das im nächsten Abschnitt beschriebene Kommando `growisofs` unterstützt nur den zweiten und den dritten Modus. Welcher Modus zum Einsatz kommt, hängt bei DVD-RW-Medien von der Formatierung ab. `dvd+rw-format` formatiert die Medien normalerweise für den Modus *restricted overwrite*.

Wenn Sie den Modus *incremental sequential* nutzen möchten, müssen Sie die Option `-blank` angeben. Das Formatieren dauert dann leider sehr lange (die

volle Spieldauer, dividiert durch den Geschwindigkeits-Faktor, also ca. eine halbe Stunde bei einem 2X-Medium). Fabrikneue DVD-RW-Rohlinge sind normalerweise für den Modus *incremental sequential* vorformatiert.

Im Regelfall ist es zweckmäßig, einfach den Default-Modus der `dvd+rw-tools` zu verwenden, also *restricted overwrite*. Eine Menge technischer Hintergründe zu den verschiedenen Formatierungsvarianten finden Sie unter *http://fy.chalmers.se/˜appro/linux/DVD+RW/*.

Informationen über eine DVD ermitteln: Wenn Sie eine DVD erhalten und nicht wissen, um welchen DVD-Typ es sich handelt, ob die DVD schon beschrieben ist und wenn ja, in welchem Modus und mit wie vielen Sessions, können Sie diese Informationen mit `dvd+rw-mediainfo` ermitteln:

```
user$  dvd+rw-mediainfo /dev/sr0
INQUIRY:                [_NEC    ][DVD_RW ND-1300A ][1.07]
GET [CURRENT] CONFIGURATION:
 Mounted Media:         1Ah, DVD+RW
GET PERFORMANCE:
 Speed Descriptor#0:    00/221280 Reading@7.8x Writing@2.3x
READ DVD STRUCTURE[#0h]:
 Media Book Type:       92h, DVD+RW book [revision 2]
 Media ID:              RICOHJPN/W01
 Legacy lead-out at:    221280*2KB=453181440
...
```

growisofs (DVD-R[W]s und DVD+R[W]s schreiben)

Das Kommando `growisofs` aus den `dvd+rw-tools` schreibt DVD+Rs, DVD+RWs, DVD-Rs und DVD-RWs. Bei DVD-R[W]s werden allerdings nur die Modi *incremental sequential* und *restricted overwrite* unterstützt. Wenn Sie DVD-RWs im Modus *disk at once* schreiben möchten, müssen Sie momentan das kommerzielle Programm `cdrecord-ProDVD` einsetzen. (Die Modi sind im vorigen Abschnitt beschrieben.)

```
growisofs [optionen] device
```

`growisofs` greift auf `mkisofs` zurück. Deswegen sind fast alle Optionen mit den Optionen von `mkisofs` identisch (siehe Seite 324). Es gibt nur wenige neue Optionen (Tabelle 7.5):

Beispiel: Das folgende Kommando speichert den Inhalt des Verzeichnisses `daten` auf einer DVD. (Die `mkisofs`-Optionen `-r` und `-J` bewirken, dass die DVD lange Dateinamen entsprechend den Rockridge- und Joliet-Erweiterungen aufweist.) Statt des Device-Namens `/dev/sr`*n* müssen Sie je nach Distribution `/dev/scd`*n* angeben.

```
user$  growisofs -r -J -Z /dev/sr0 daten/
```

Tabelle 7.5: Optionen von growisofs

-Z *device*	Diese Option gibt das DVD-Device an. Falls Sie Multi-Session-DVDs erzeugen möchten, schreiben Sie die erste Session mit dieser Option.
-M *device*	Bei Multi-Session-DVDs muss *-M* ab der zweiten Session verwendet werden. `growisofs` kümmert sich selbstständig darum, dass `mkisofs` die Session an der korrekten Position beginnt. Achten Sie darauf, dass Sie bei Multi-Sessions immer dieselben *mkisofs*-Optionen angeben. Das gilt insbesondere für alle Optionen, die einen Einfluss auf das ISO-Dateisystem haben (Joliet- und Rockridge-Erweiterungen etc.).
-dry-run	Mit dieser Option führt `growisofs` einen Probelauf durch, ohne tatsächlich Daten zu schreiben.
-dvd-compat	Die Option erhöht die Kompatibilität zu DVD-ROM-Laufwerken. DVD-Rs bzw. DVD+Rs werden abgeschlossen (und können daher nicht durch weitere Sessions erweitert werden). Bei DVD+RWs wird nach den Daten der (an sich optionale) *Lead-out*-Bereich geschrieben.
-speed=n	Die Option gibt die gewünschte Schreibgeschwindigkeit an. Das ist normalerweise nicht notwendig – das Laufwerk bestimmt selbst die korrekte Geschwindigkeit. Sie können aber durch *-speed=1* ein langsameres Brennen erzwingen.

Eine zweite Session fügen Sie so hinzu:

```
user$  growisofs -r -J -Z /dev/sr0 nochmehrdaten/
```

Manche DVD-ROM-Laufwerke kommen generell nicht mit DVD+RWs zurecht (egal, ob Single- oder Multi-Session). Manchmal gelingt es folgendermaßen, die Probleme zu umgehen. Man schreibt einen *Lead-out*-Bereich auf die DVD+RW und verändert anschließend die *Book-type*-Information der DVD:

```
user$  dvd+rw-format -lead-out /dev/sr0
user$  dvd+rw-booktype -dvd-rom -media /dev/sr0
```

Normalerweise übergibt `growisofs` alle Optionen außer `-Z` bzw. `-M` an `mkisofs` und schreibt das Ergebnis von `mkisofs` direkt auf die DVD. Wenn Sie ein bereits existierendes ISO-Image schreiben möchten, lautet die Syntax *-Z device=isodatei*:

```
user$  growisofs -Z /dev/sr0=daten.iso
```

Kapitel 8

Textbearbeitung mit LaTeX und LyX

Einführung

LaTeX ist ein System zum Setzen (Layouten) von Texten. Sie können damit vom Brief bis hin zu einem Buch beinahe jeden beliebigen Text gestalten. LaTeX ist wegen seines herausragenden Formelsatzes vor allem in der (natur-)wissenschaftlichen Welt sehr beliebt. Zahllose Diplomarbeiten, Dissertationen und wissenschaftliche Veröffentlichungen wurden und werden damit verfasst – so auch dieses Buch.
Dieses Kapitel bietet einen Schnelleinstieg in LaTeX und fasst die wichtigsten LaTeX-Anweisungen zusammen. Außerdem finden Sie einige Hintergrundinformationen darüber, wie LaTeX Schriften nutzt, wie Sie aus LaTeX-Dokumenten PostScript- und PDF-Dateien erzeugen etc. Der Platz gestattet es leider nicht, auf die vielen vorzüglichen Makripakete einzugehen, die es für nahezu jeden Zweck gibt – hier müssen wir auf die weiterführende Literatur und den Server von Dante e. V. verweisen.
Wenn Sie die Vorteile von LaTeX nutzen möchten, ohne den größten Nachteil – die schwierige Bedienung – in Kauf zu nehmen, stellt LyX eine attraktive Variante dar. LyX ist eine Art Benutzeroberfläche zu LaTeX, die beinahe WYSIWYG realisiert.

8.1 Einführung

Die besonderen Vorzüge von LaTeX gegenüber anderen Programmen bestehen im grandiosen Formelsatz ($\pi \sum_{i=1}^{n} \frac{a_i x^i}{i!}$) und in der überragenden Satzqualität (automatischer Zeichenausgleich). Beispielsweise wird in „Vektor“ das „e“ näher an das „V“ gerückt. LaTeX hat aber auch Nachteile: Die Bedienung des Programms

ist steinzeitlich. Scheinbar triviale Dinge wie manche Trennungen oder Seitenumbrüche müssen oft manuell verändert werden, wenn LATEX standardmäßig nicht die gewünschten Resultate liefert.

Kurz einige Worte zur Herkunft von TEX und LATEX: Genau genommen ist LATEX ein Makropaket, das das Satzprogramm TEX erweitert. TEX ist aber noch komplizierter anzuwenden als LATEX und steht deswegen im Schatten seiner Erweiterung LATEX.

TEX wurde in seiner ursprünglichen Form von Donald Knuth programmiert, das dazugehörige Makropaket LATEX von Leslie Lamport. Seit Leslie Lamport die Weiterentwicklung von LATEX mit Version 2.09 eingestellt hat, sind es vor allem Frank Mittelbach und Rainer Schöpf, denen die aktuelle Version LATEX 2_ε und die Pläne zur Weiterentwicklung in Richtung LATEX 3 zu verdanken sind.

LATEX von Leslie Lamport ist zwar im Vergleich zu TEX relativ einfach zu erlernen, es kann aber nicht mit dem Komfort gewöhnlicher Textverarbeitungsprogramme wie OpenOffice Writer, KWord oder Microsoft Word mithalten – was das WYSIWYG (*What You See Is What You Get*) betrifft, denn bei LATEX sieht man zunächst nur Text mit Befehlen zur späteren Struktur und Formatierung. Das Generieren des Layouts erfolgt in einem gesonderten Schritt (beinahe so wie bei der Erstellung eines Programm-Quelltextes und der späteren Kompilierung). Dafür können die Hände beim Schreiben auf der Tastatur bleiben. Bei LATEX handelt es sich also eher um ein Programm vom Typ WYSIWYM (*What You See Is What You Mean*) – und das ist sehr viel besser.

Der größte Vorteil von LATEX liegt jwedoch darin, dass der Dokumenten-Quelltext in reinem ASCII erzeugt wird und man so auch Dokumente noch verwerten kann, die älter sind. (Versuchen Sie mal ein zehn Jahre altes Word-Dokument zu bearbeiten.) Außerdem ist LATEX plattformunabhängig – man kann die Dokumente problemlos zwischen Linux, Windows, Mac OS oder anderen Betriebssystem- und Rechnerplattformen austauschen. Aber wir geben auch zu, dass die Arbeit mit LATEX nach den ersten leichten Schritten fummelig werden kann, etwa wenn es um Tabellen oder ein bestimmtes Layout geht. Dafür werden auch Bücher mit 1200 Seiten und zahllosen Abbildungen ohne Murren verarbeitet.

Für die Super-Profis gibt es noch einen weiteren Vorteil. Man kann LATEX-Code auch per Programm erzeugen und so beispielsweise die Ergebnisse einer Datenbankabfrage in LATEX-Code einfügen und daraus ein supertolles PDF-Dokument erzeugen – und das ohne jeden manuellen Eingriff.

Weiterführende Dokumentation: Dieses Kapitel kann nur eine LATEX-Einführung geben. Darüber hinausgehende Informationen finden Sie in den zahlreichen LATEX-Büchern. Besonders zu empfehlen sind dabei die Bücher von Helmut Kopka und der *LATEX-Begleiter* von Michael Goosens, Frank Mittelbach und Alexander Samarin.

Zusammen mit LaTeX sind eine Menge Dokumentationsdateien installiert, bei aktuellen Versionen überwiegend im PDF-Format. Entsprechende Dateien finden Sie mit den folgenden Kommandos:

```
user$ find /usr/share/texmf -name '*.pdf'
user$ locate '/usr/share/texmf/*.pdf'
```

Sehr hilfreich sind auch die folgenden Seiten im Internet:

http://www.dante.de/faq/de-tex-faq/
http://www.ctan.org/tex-archive/info/german/
http://www.ctan.org/tex-archive/info/german/LaTeX2e-Kurzbeschreibung/l2kurz2.pdf

8.1.1 LaTeX und die wichtigsten Hilfsprogramme

LaTeX ist ein Satzprogramm und kein Textverarbeitungsprogramm. Der Unterschied besteht darin, dass LaTeX nicht mit einem eigenen Editor ausgestattet ist. Vielmehr muss der zu setzende Text als gewöhnliche Textdatei mit einem beliebigen Editor geschrieben werden. Daraus ergibt sich auch, dass LaTeX kein WYSIWYG-Programm ist – ganz im Gegenteil: Sämtliche Satzanweisungen müssen in einer ziemlich unübersichtlichen Syntax im Text angegeben werden. Wenn Sie beispielsweise ein Wort kleinschreiben möchten, lautet die LaTeX-Syntax hierfür `{\small klein}`.

Zur Texteingabe können Sie grundsätzlich jeden beliebigen Editor verwenden. Besonders gut geeignet sind natürlich Editoren, die LaTeX verstehen, LaTeX-Schlüsselwörter farbig hervorheben und eventuell auch bei der Übersetzung der LaTeX-Datei und der Fehlersuche behilflich sind (z. B. Emacs, kate). Noch mehr Komfort bieten LaTeX-Umgebungen wie das Programm `kile` (ehemals `ktexmaker`).

Der nächste Schritt nach der Texteingabe besteht darin, aus der LaTeX-Datei mit dem Kommando `latex name.tex` eine DVI-Datei zu erstellen (Kennung `*.dvi`). Dabei handelt es sich um eine Datei, in der alle Anweisungen für das Seitenlayout in einer drucker- bzw. device-unabhängigen Sprache angegeben werden.

Bei der Ausführung von `latex` kommt es häufig zu Fehlermeldungen, die auf Syntaxfehler in der LaTeX-Datei zurückzuführen sind. Einige Informationen zum Umgang mit Fehlermeldungen und zur Fehlersuche finden Sie auf Seite 339.

Sobald die DVI-Datei vorliegt, kann sie mit den Programmen `xdvi` oder `kdvi` betrachtet werden. Wenn Sie die Datei ausdrucken möchten, ist noch ein weiterer Arbeitsschritt erforderlich – die Umwandlung der DVI-Datei in das PostScript-Format. Für diese Umwandlung ist das Programm `dvips` zuständig (Seite 423).

PostScript-Dateien können je nach Distribution mit `ghostview`, `gv`, `ggv` oder `kghostview` am Bildschirm betrachtet werden. Falls Sie Ihren Drucker korrekt konfiguriert haben, können Sie die PostScript-Datei natürlich auch ausdrucken.

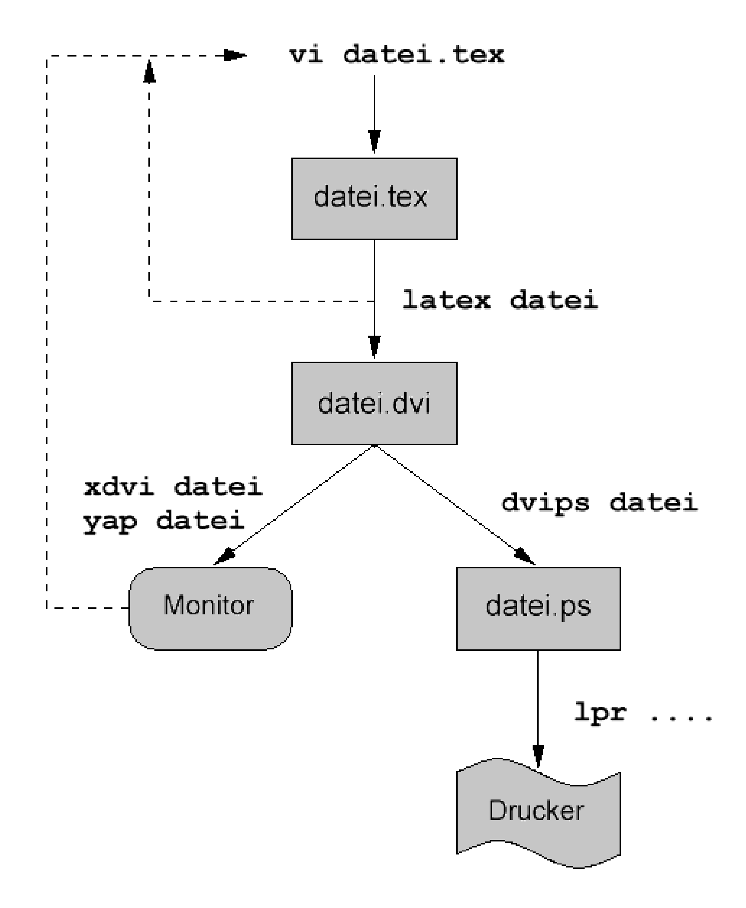


Abbildung 8.1: Prinzipieller Ablauf beim Erstellen eines LaTeX-Dokuments

Sie können aber auch weitermachen und eine PDF-Datei erzeugen. Im Einzelnen sieht der Weg von der Textdatei `test.tex` im LaTeX-Format bis hin zum Ausdruck folgendermaßen aus:

```
user$ latex test.tex                #liefert test.dvi
user$ dvips -o test.ps test.dvi     #liefert test.ps
user$ ghostview test.ps &           #Ergebnis überprüfen
user$ lpr test.ps                   #Ausdruck (oder auch .pdf generieren)
```

Beim ersten Ausführen von `dvips` oder `xdvi` werden neue Schriftartdateien generiert. Die beiden Kommandos starten dazu automatisch das Programm `mf` (Metafont). Gegebenenfalls muss auch der `latex`-Durchlauf zweimal (in seltenen Fällen dreimal) erfolgen, damit alle Querverweise korrekt abgesättigt werden. Programmierprofis basteln sich meist ein `Makefile`, das den Anlauf automatisiert, zum Beispiel:

```
BUCH = linux

all:
	@echo "Aufruf : make <Option>"
	@echo
	@echo "Gueltige Optionen sind :"
	@echo "dvi   - Erzeugt Latex-Lauf und Anzeige ueber xdvi"
	@echo "ps    - Latex-Lauf und anschliessend dvips"
	@echo

dvi:
	@rm -f ${BUCH}.log missfont.log
	@latex ${BUCH}
	@makeindex -s myindex.sty ${BUCH}
	@latex ${BUCH}
	@makeindex -s myindex.sty ${BUCH}
	@xdvi ${BUCH}

ps:
	@rm -f ${BUCH}.log missfont.log
	@latex ${BUCH}
	@makeindex -s myindex.sty ${BUCH}
	@latex ${BUCH}
	@makeindex -s myindex.sty ${BUCH}
	@dvips -t a4 -o ${BUCH}.ps ${BUCH}.dvi
	@ghostview test.ps

coffee:
	@echo
	@echo "With milk and sugar?"
```

Wer keine Make-Dateien mag, kann auch ein Shell-Programm mit gleicher Funktion schreiben. Generell gilt, dass man sich nicht erst mit den Anfängen von LaTeX befassen sollte, wenn der Abgabetermin für Artikel, Diplomarbeit oder was auch immer schon in greifbare Nähe gerückt ist.

8.1.2 Fehlersuche in LaTeX-Texten

Der erste Kontakt mit dem Programm LaTeX ist in der Regel frustrierend. Das Programm arbeitet interaktiv, reagiert auf den ersten auftretenden Fehler mit einer fast immer unverständlichen Fehlermeldung und erwartet dann auch noch von Ihnen, dass Sie durch die Eingabe eines Buchstabens angeben, wie es weitergehen soll. Eine typische Fehlermeldung sieht beispielsweise so aus:

```
LaTeX error.  See LaTeX manual for explanation.
              Type  H <return>  for immediate help.
! Text for \verb command ended by end of line.
\@latexerr ...rcontextlines \m@ne \errmessage {#1}
```

```
                                                   \endgroup
l.46 ...das folgende Kommando {\verb?{\small
?
```

Die Fehlerursache ist in diesem Fall ein `\verb`-Kommando in Zeile 46, dessen Wirkung über das Ende dieser Zeile hinausreicht (und das ist nicht erlaubt). Sie haben jetzt folgende Möglichkeiten, LaTeX fortzusetzen:

- ⏎ setzt die Verarbeitung der LaTeX-Datei ohne Rücksicht auf den gerade aufgetretenen Fehler fort. Manchmal funktioniert das, sehr häufig führt es zu zahlreichen Folgefehlern (auf die ebenfalls mit ⏎ reagiert werden kann).
- H ⏎ zeigt zusätzliche Informationen zur Fehlermeldung an. Der Infotext ist allerdings nur in den seltensten Fällen eine echte Hilfe.
- R ⏎ setzt die Verarbeitung fort, zeigt weitere Fehlermeldungen an, erwartet aber keine Eingaben mehr.
- Q ⏎ wie oben, aber ohne die Anzeige von Fehlermeldungen
- X ⏎ beendet LaTeX.

In der Regel werden Sie LaTeX entweder mit X sofort beenden, wenn Sie die Fehlerursache erkannt haben, oder die Bearbeitung mit Q im Quiet-Modus fortsetzen. Im zweiten Fall können Sie darauf hoffen, dass LaTeX trotz der höchstwahrscheinlich auftretenden Folgefehler in der Lage ist, zumindest die beanstandete Seite fertig zu übersetzen. In diesem Fall können Sie mit `xdvi` das Ergebnis (die gesetzte Seite) ansehen und dort vielleicht die Fehlerursache erkennen.

In jedem Fall werden Sie anschließend in den Editor wechseln und dort den Fehler in der LaTeX-Datei suchen. Dabei ist die Datei `name.log` eine wesentliche Hilfe. In dieser Datei werden alle Fehlermeldungen von LaTeX gespeichert (unabhängig davon, ob sie auf dem Bildschirm angezeigt wurden oder nicht). Der Dateiname dieser Protokolldatei setzt sich aus dem Namen der übersetzten LaTeX-Datei und der Kennung `.log` zusammen. Die wichtigste Information in dieser Datei ist in der Regel die Nummer der Zeile, in der der Fehler aufgetreten ist.

Wenn Sie Probleme beim Aufspüren eines Fehlers haben, sollten Sie versuchen, den kritischen Textausschnitt zu isolieren und in eine eigene, möglichst kleine Datei zu kopieren. Generell empfiehlt sich bei umfangreichen Texten eine Zerlegung in mehrere Dateien.

LaTeX liefert während der Bearbeitung von Texten nicht nur Fehlermeldungen, sondern auch Warnungen. Bei Warnungen wird die Bearbeitung des Textes nicht unterbrochen. Viele Warnungen beginnen zumeist mit einem Text wie *overfull hbox* und deuten darauf hin, dass LaTeX Probleme beim Zeilen- oder Seitenumbruch hat. In solchen Fällen sind zumeist manuelle Eingriffe im Text erforderlich

(Trennvorschläge, erzwungene Seitenumbrüche etc.). Oft ist LaTeX hierbei aber auch zu pingelig.

Abschließend einige Tipps und Hinweise zum richtigen Umgang mit LaTeX, wenn Probleme auftreten:

- Das interaktive Verhalten von LaTeX – dass also bei jedem Fehler eine Unterbrechung auftritt – kann sehr lästig sein, vor allem, wenn die Übersetzung automatisiert werden soll. Wenn Sie am Beginn der LaTeX-Datei die Anweisung `batchmode` einfügen, arbeitet LaTeX auch bei Fehlern interaktiv (so, als würde beim ersten Fehler (Q) eingegeben). Nach dem Ende der Übersetzung können Sie in Ruhe die `*.log`-Datei lesen.
- Während LaTeX eine Tastatureingabe erwartet, reagiert das Programm nicht auf (Strg)+(C)! Wenn Sie das Programm während einer Eingabe beenden möchten, müssen Sie (Strg)+(D) (für End of File) drücken! Dieser Notausstieg ist insbesondere dann praktisch, wenn LaTeX auf einen falschen Dateinamen gestoßen ist und von Ihnen die Angabe einer anderen Datei erwartet. Einen alternativen Ausweg stellt die Eingabe von `null` dar. LaTeX lädt dann die leere Datei `null.tex` bzw. `null.sty`, die extra für diesen Zweck in der LaTeX-Verzeichnisstruktur vorgesehen ist.
- In seltenen Fällen steckt der Fehler nicht in der zu übersetzenden LaTeX-Datei, sondern in der Datei für das Inhaltsverzeichnis, die beim vorangegangenen Durchlauf erzeugt worden ist. Löschen Sie die Datei `name.toc`!
- Wenn bei der Übersetzung einer fremden `*.tex`-Datei zahlreiche unerklärliche Fehlermeldungen auftreten, dann liegt das zumeist daran, dass es sich nicht um eine LaTeX-, sondern um eine TeX-Datei handelt. TeX-Dateien weisen ebenfalls die Kennung `*.tex` auf, müssen aber mit `tex dateiname.tex` übersetzt werden!

Ein praktisches Hilfsmittel bei der Fehlersuche ist das Programm `lacheck`. Es analysiert die als Parameter übergebene LaTeX-Datei und liefert eine Liste mit Warnungen über mögliche Fehler.

8.1.3 Einführungsbeispiel

Bevor der nächste Abschnitt eine systematische Beschreibung der wichtigsten LaTeX-Kommandos liefert, soll das folgende Beispiel den prinzipiellen Umgang mit LaTeX demonstrieren. Die unten abgedruckten LaTeX-Anweisungen ergeben nach der Übersetzung durch LaTeX die in der Abbildung dargestellte Seite. Bei einem längeren Artikel wäre es natürlich sinnvoll, das Inhaltsverzeichnis auf einer eigenen Seite darzustellen und dem ganzen Artikel eine Titelseite voranzustellen – darauf wurde hier aus Platzgründen verzichtet.

Inhaltsverzeichnis

1 LaTeX-Einführung

1.1 Gestaltung des Schriftbilds

Dieser Text zeigt einige Gestaltungsmöglichkeiten in LaTeX: **fette Schrift**, *kursive Schrift*, KAPITÄLCHEN, Sans Serif, `Typewriter`. Neu in LaTeX 2_ε ist die Tatsache, dass sich Schriftattribute jetzt weitgehend problemlos kombinieren lassen – beispielsweise ***fett und kursiv***. Natürlich kann auch die Schriftgröße verändert werden von ganz winzig über klein bis ziemlich groß.

1.2 Textblöcke und Rahmen

Mit der `minipage`-Umgebung können Textblöcke nebeneinander angeordnet werden.

Das ist die zweite, etwas schmalere Minipage.

Hier wurde eine 5 cm breite Minipage durch ein vor- und ein nachgestelltes `\hfill`-Kommando zentriert und mit `\fbox` eingerahmt.

1.3 Aufzählungen

LaTeX hat viele Vorteile gegenüber anderen Programmen:

- Die Qualität der Ergebnisse spricht für sich.
- Die Verarbeitungsgeschwindigkeit ist sehr hoch, wenn man sich an die Syntax gewöhnt hat.
- LaTeX-Texte sind portabel und werden in der Unix-Welt oft zur Online-Dokumentation eingesetzt.

1.4 Fußnoten

Dieser Absatz liefert zwei Beispiele für Fußnoten.[1] LaTeX nummeriert die Fußnoten[2] natürlich automatisch.

1.5 Mathematische Formeln

Seine noch immer große Bedeutung verdankt LaTeX in erster Linie seinem hervorragenden Formelsatz. Versuchen Sie, die folgenden Formeln einmal in einem anderen Programm einzugeben! Formeln können übrigens auch direkt im Text (etwa hier: $\pi \int x^2 dx$) verwendet werden.

$$\lim_{n\to\infty} \sum_{k=1}^{n} \sqrt[3]{1+\frac{k^2}{n^3}} - n$$

$$\begin{bmatrix} \frac{\partial}{\partial x} f & \frac{\partial}{\partial y} f \\ \frac{\partial}{\partial x} g & \frac{\partial}{\partial y} g \end{bmatrix}$$

[1] Das ist die erste Fußnote.
[2] Die zweite Fußnote.

1

Abbildung 8.2: Der von LaTeX bearbeitete Text `article.tex`

```
% Dokumenttyp: zweispaltiger Artikel
\documentclass[a4paper,twocolumn,11pt]{article}
\usepackage{ngermeman}          % deutsche Überschriften
\usepackage[latin1]{inputenc}    % Latin-1-Zeichensatz
\parindent 0pt                   % kein Einrücken der ersten Zeile
\parskip1ex                      % Leerraum zw. Absätzen
\columnsep 1cm                   % 1 cm Abstand zwischen den Spalten
\begin{document}                 % Beginn des eigentlichen Textes
\tableofcontents                 % Inhaltsverzeichnis einfügen

\section{\LaTeX-Einführung}

\subsection{Gestaltung des Schriftbilds}

Dieser Text zeigt einige Gestaltungsmöglichkeiten in \LaTeX:
\textbf{fette Schrift}, \textit{kursive Schrift},
\textsc{Kapitälchen}, \textsf{Sans Serif}, \texttt{Typewriter}. Neu in
\LaTeXe\ ist die Tatsache, dass sich Schriftattribute jetzt weitgehend
problemlos kombinieren lassen -- beispielsweise \textbf{\textit{fett
und kursiv}}. Natürlich kann auch die Schriftgröße verändert werden
von {\tiny ganz winzig} über {\small klein} bis {\Large ziemlich
groß}.

\subsection{Textblöcke und Rahmen}

{\small
\begin{minipage}[t]{4cm}
Mit der {\verb?minipage?}-Um\-ge\-bung können Textblöcke
nebeneinander angeordnet werden.
\end{minipage}
\hfill
\begin{minipage}[t]{3cm}
Das ist die zweite, etwas schmalere Minipage.
\end{minipage}
}

\hbox{}\hfill\fbox{
\begin{minipage}{5cm}
Hier wurde eine 5 cm breite Mini\-page durch ein vor- und ein
nachgestelltes {\small\tt \textbackslash{}hfill}-Kom\-man\-do zentriert und mit
{\small\tt \textbackslash{}fbox} eingerahmt.
\end{minipage}
}\hfill\hbox{}

\subsection{Aufzählungen}

\LaTeX\ hat viele Vorteile gegenüber anderen Programmen:

\begin{itemize}
\item Die Qualität der Ergebnisse spricht für sich.
```

```
\item Die Verarbeitungsgeschwindigkeit ist sehr hoch, wenn man sich
      an die Syntax gewöhnt hat.
\item \LaTeX-Texte sind portabel und werden in der Unix-Welt oft
      zur Online-Dokumentation eingesetzt.
\end{itemize}

\subsection{Fußnoten}

Dieser Absatz liefert zwei Beispiele für Fuß\-noten.\footnote{Das ist
die erste Fußnote.} \LaTeX\ nummeriert die Fuß\-noten\footnote{Die
zweite Fußnote.} natürlich automatisch.

\subsection{Mathematische Formeln}

Seine noch immer große Bedeutung verdankt \LaTeX\ in erster Linie
seinem hervorragenden Formelsatz.  Versuchen Sie, die folgenden
Formeln einmal in einem anderen Programm einzugeben! Formeln können
übrigens auch direkt im Text (etwa hier: $\pi \, \int x^2 dx$)
verwendet werden.

\[\lim _{n\rightarrow \infty }\sum _{k=1}^{n}\sqrt
  [3]{1+{\frac {k^{2}}{n^{3}}}}-n\]

\[\left [\begin {array}{cc}
{\frac {\partial }{\partial x}}f&{\frac {\partial }{\partial y}}f\\
\noalign{\medskip}
{\frac {\partial }{\partial x}}g&{\frac {\partial }{\partial y}}g
\end {array}\right ]\]
\end{document}
```

Wenn Sie nach dem Studium dieses Abschnitts den Eindruck gewonnen haben, dass die Arbeit mit LaTeX eine manchmal etwas mühselige Angelegenheit ist, haben Sie prinzipiell Recht. Der Weg bis zu einem wirklich optimalen Ergebnis ist oft dornig. Zum Teil sind manuelle Eingriffe notwendig, damit LaTeX deutsche Wörter richtig trennt und Seiten dort umbricht, wo es aus ästhetischen Gründen sinnvoll ist. Jedoch spricht die Qualität von LaTeX-Texten für sich.

Im mathematisch-naturwissenschaftlichen Sektor gibt es zudem keine ernsthafte Alternative zu LaTeX. Textverarbeitungsprogramme wie Microsoft Word oder OpenOffice Writer sind für umfangreiche wissenschaftliche Texte mit Formeln und Bildern zu ineffizient und oft auch zu instabil. Was nützt eine tolle Benutzeroberfläche, wenn ein Textverarbeitungsprogramm bei längeren Texten plötzlich abstürzt, Querverweise nicht mehr findet, statt Abbildungen nur noch ein rotes „X" anzeigt? Der Einstieg in LaTeX dauert sicherlich ein paar Tage länger als bei anderen Programmen – aber diese Zeit holen Sie später wieder auf, wenn das Programm auch mit hundert- oder tausendseitigen Dokumenten noch problemlos funktioniert! Wie auch bei anderen Programmen können Sie sich für LaTeX Formatvorlagen erstellen, die dann nur noch ausgefüllt werden müssen. Berühmt ist es jedoch für seine Erweiterbarkeit. Es lassen sich jederzeit neue

Makros oder komplette Pakete erstellen, die dann die Arbeit erleichtern. So handelt es sich beispielsweise bei den Tastensymbolen in diesem Buch um ein Makro *keys*, das mit dem Aufruf `\keys{Drück mich!}` das Ergebnis (Drück mich!) hat. So lassen sich auch Kästen und andere Elemente austüfteln und dann sehr einfach verwenden.

8.1.4 teTeX

Das Programm LaTeX kann nicht isoliert gesehen werden, sondern muss in seiner Gesamtheit mit diversen Style-Dateien, dem Zusatzprogramm Metafont, dessen Schriftarten etc. betrachtet werden. Diese Gesamtheit wird als „LaTeX-Distribution" bezeichnet.

Die zurzeit am weitesten verbreitete LaTeX-Distribution ist das von Thomas Esser gewartete teTeX (*http://www.tug.org/teTeX/*). Alle Aussagen in diesem Kapitel beziehen sich auf die aktuelle teTeX-Version 2.0. Da nur wenige Anwender wirklich alle TeX- und LaTeX-Erweiterungen benötigen, wird teTeX bei den meisten Linux-Distributionen in mehrere Pakete zerlegt, von denen per Default meist nur wenige installiert werden. Dabei werden in der Regel folgende Verzeichnisse genutzt:

`/etc/texmf/`	Konfigurationsdateien
`/usr/share/texmf/`	TeX- und LaTeX-Dateien
`/var/lib/texmf`	veränderliche Dateien
`/var/cache/fonts/`	dynamisch erzeugte Font-Bitmaps (*.pk)

Angesichts der unüberschaubaren Anzahl von Dateien kann die Suche nach einer bestimmten Font- oder Style-Datei relativ lange dauern. Bei der teTeX-Distribution wird daher eine zusätzliche Datenbank verwaltet, die eine Liste aller TeX-Dateien enthält. Die Datenbank besteht im Wesentlichen aus dem Ergebnis von `ls -R` und hat daher den Dateinamen `ls-R`.

Falls Sie teTeX manuell um zusätzliche Dateien erweitern, müssen Sie unbedingt das Kommando `texhash` ausführen, damit die Datenbank `ls-R` aktualisiert wird. Auch sonst sollte dieses Kommando die erste Maßnahme sein, wenn es Probleme beim Auffinden von TeX-Dateien gibt. Intern ist die Kpathsea-Library für die Suche nach TeX-Dateien zuständig. Wenn die Details Sie interessieren, lesen Sie `kpathsea.dvi` oder die äquivalenten `info`-Texte sowie die sehr kurzen `man`-Pages zu `texhash` und `ls-R`.

In der teTeX-Distribution sind zwar schon sehr viele Pakete und Werkzeuge vereint, aber es gibt noch weit mehr LaTeX-Utilities, -Erweiterungen und -Speziallayouts (etwa zum Dokumentieren von Schachstellungen, zum Notensatz etc.). Diese Erweiterungen werden vom CTAN (*Comprehensive TeX Archive Network*) gesammelt:

http://www.latex-project.org/
http://www.dante.de
http://dante.ctan.org/CTAN

8.2 Elementare LaTeX-Kommandos

Dieser Abschnitt fasst die wichtigsten LaTeX-Kommandos zusammen. Diese Kommandos sollten ausreichen, um einfache LaTeX-Dokumente zu erstellen. Wenn Sie intensiver mit LaTeX arbeiten möchten, führt aber an weiterführender Literatur kein Weg vorbei.

8.2.1 Formale Details

Die Arbeit mit LaTeX beginnt in einem beliebigen Texteditor. Im Text gilt eine Gruppe zusammenhängender Zeilen als Absatz. Wörter werden durch Leerzeichen oder Zeilenumbrüche voneinander getrennt, Absätze durch mindestens eine Leerzeile. Der eigentliche Zeilenumbruch innerhalb eines Absatzes wird von LaTeX durchgeführt. Deswegen spielt es keine Rolle, an welcher Stelle im Ausgangstext eine neue Zeile begonnen wird.

LaTeX-Kommandos (genauer: der Aufruf eines LaTeX-Makros) beginnen immer mit einem Backslash `\`. Wenn die Kommandos Parameter benötigen, stehen diese in geschweiften Klammern, also beispielsweise `\chapter{Kapitelüberschrift}`. Manche Kommandos kennen auch optionale Parameter, die in eckigen Klammern angegeben werden, z. B. `$\sqrt[3]{x}$` für $\sqrt[3]{x}$. Ohne `[3]` liefert `\sqrt` eine gewöhnliche Quadratwurzel, also $\sqrt{x}$. Geschweifte Klammern können auch dazu verwendet werden, die Wirksamkeit von Kommandos einzuschränken. So wird durch `{\bf Wort}` nur ein einziges Wort fett gedruckt, während das ungeklammerte Kommando `\bf` das Attribut „Fett" bis auf Widerruf einstellt. Manchmal dienen sie auch zur Trennung vom nachfolgenden Text, wenn sonst das Makro nicht erkannt würde (z. B. `\textbackslash{}LaTeX` statt `\textbackslashLaTeX`).

Umgebungen stellen einen besonderen Typ von Kommandos dar. Sie werden mit `\begin{umgebung}` eingeleitet und mit `\end{umgebung}` abgeschlossen. Für den gesamten Text zwischen diesen beiden Kommandos gelten die besonderen Formatierungsmerkmale der Umgebung. Typische Umgebungsnamen sind *tabbing* (für Tabellen) oder *verbatim* für Listings mit Sonderzeichen.

Innerhalb des LaTeX-Textes können mit `%` Kommentare eingeleitet werden. Der Rest der Zeile ab diesem Zeichen wird von LaTeX nicht beachtet.

8.2.2 Vorspann

LaTeX-Texte beginnen mit dem Kommando `\documentclass[optionen]{typ}`. Dieses Kommando bestimmt den Texttyp. LaTeX kennt in der Standardkonfiguration vier wichtige Texttypen: *book*, *report*, *article* und *letter*. Die drei ersten Texttypen sind einander relativ ähnlich und unterscheiden sich primär in der vorgesehenen Textlänge. *article* kennt im Gegensatz zu *book* und *report* keine Kapitel; die kleinste Gliederungseinheit ist dort ein Abschnitt (*section*). In *book* werden alle Seiten automatisch mit Kopfzeilen ausgestattet, in denen neben der Seitennummer auch der Name des aktuellen Kapitels (gerade Seiten) und der Name des aktuellen Abschnitts (ungerade Seiten) angegeben wird. Am leichtesten erkennen Sie die Unterschiede zwischen den drei Texttypen *book*, *report* und *article*, wenn Sie die erste Zeile der Beispieldatei des vorangegangenen Abschnitts ändern und dort der Reihe nach alle drei Typen einsetzen.

Der Texttyp *letter* kann zum Verfassen von Briefen verwendet werden. Auf diesen Texttyp wird hier aus Platzgründen allerdings nicht eingegangen. LaTeX-intern werden Texttypen durch Makrodateien mit der Kennung `*.cls` realisiert. Diese Dateien sind im Verzeichnis `/usr/share/texmf/tex/latex/base` gespeichert. Manche Verlage, Universitäten etc. stellen darüber hinaus eigene Makrodateien zur Verfügung, die sich speziell zur Formatierung von wissenschaftlichen Artikeln, Diplomarbeiten etc. eignen.

Vor dem Texttyp können in eckigen Klammern Optionen angegeben werden. Wichtige Optionen sind `11pt` und `12pt` (sie verändern die Standardschriftgröße), *a4paper* (DIN-A4-Papierformat) und `twocolumn` (zweispaltige Texte).

LaTeX-Dokumentklassen

`\documentclass[optionen]{typ}`

Texttypen

`book`	für lange Texte (Bücher), Gliederung in Teile, Kapitel, Abschnitte
`report`	wie oben, aber für kürzere Texte; andere Titelseite etc.
`article`	für Artikel, Gliederung in Abschnitte, Unterabschnitte; im Gegensatz zu *book* und *report* keine Unterscheidung zwischen geraden und ungeraden Seitenzahlen
`letter`	für Briefe (Für uns besser geeignet ist `dinbrief`.)

Optionen

`11pt`	Standardschriftgröße 11 (statt 10) Punkt
`12pt`	Standardschriftgröße 12 (statt 10) Punkt
`a4paper`	DIN-A4-Format (statt US-Letter)
`twoside`	Unterscheidung gerade/ungerade Seite (Standard bei *book*)
`twocolumn`	zweispaltiger Text

Tipp

Statt *book*, *report*, *article* und *letter* können Sie auch *scrbook*, *scrreprt*, *scrartcl* bzw. *scrlttr2* verwenden. Diese Formatvorlagen sind besser an die DIN-Papierformate angepasst. Außerdem werden alle Überschriften in Sans-Serif-Schriften ausgeführt. Diese Formatvorlagen sind Teil des KOMA-Pakets, das üblicherweise als Teil der teTeX-Distribution installiert wird. KOMA ist das Kürzel für Markus Kohm, der die Dateien zusammengestellt hat.

Natürlich kann man die Standardvorgaben für die Dokumentklassen jederzeit nach eigenen Wünschen ändern. LaTeX definiert jeden Teil einer Seite mit entsprechenden Werten (siehe Abbildung), die sich den aktuellen Gegebenheiten anpassen lassen. Das Setzen der neuen Werte erfolgt mittels `\setlength`, z. B.:

```
\setlength\paperwidth       {297mm}
\setlength\paperheight      {187mm}

\setlength\textheight       {198mm}
\setlength\textwidth        {126mm}
\setlength\marginparwidth   {32mm}
\setlength\marginparsep     {3.5mm}
\setlength\oddsidemargin    {4.6mm}
\setlength\evensidemargin   {4.6mm}
\setlength\topmargin        {-11.9mm}
```

Normalerweise kann man die Angaben aber so lassen, wie sie von LaTeX bzw. TeX vorgegeben worden sind.

Diverse LaTeX-Zusatzfunktionen können durch weitere Pakete aktiviert werden. Dabei handelt es sich um LaTeX-Dateien mit der Kennung `*.sty`, die diverse Systemeinstellungen verändern und zusätzliche Kommandos zur Verfügung stellen. Zusatzpakete werden mit `\usepackage[optionen]{name}` geladen. Das bzw. die `\usepackage`-Kommandos müssen unmittelbar nach `\documentclass` angegeben werden. Die folgende Tabelle zählt nur die wichtigsten Zusatzpakete auf. (Der Platz in diesem Kapitel reicht leider nicht aus, um alle Pakete zu beschreiben.)

Zusatzpakete

`\usepackage{alltt}`	Variante zur `verbatim`-Umgebung
`\usepackage{babel}`	Mehrsprachige Dokumente
`\usepackage{color}`	Farben nutzen
`\usepackage{fancyhdr}`	Gestaltung der Kopf- und Fußzeilen
`\usepackage[T1]{fontenc}`	Andere Zeichenkodierung
`\usepackage{german}`	Deutsche Trennungen, Überschriften etc.
`\usepackage{graphicx}`	Einbindung von Grafikdateien
`\usepackage{hyperref}`	PDF- und HTML-Funktionen
`\usepackage[latin1]{inputenc}`	Latin-1-Zeichensatz
`\usepackage[utf8x]{inputenc}`	UTF-8-Zeichensatz verwenden
`\usepackage{makeidx}`	Stichwortverzeichnis
`\usepackage{ngerman}`	Wie `german`, aber neue Trennregeln
`\usepackage[sf]{titlesec}`	Sans-Serif-Überschriften

LaTeX erwartet den Quelltext per Default als ASCII-Datei. Wenn Sie in der LaTeX-Datei die westeuropäischen Sonderzeichen des Latin-1-Zeichensatzes verwenden möchten, müssen Sie am Beginn des LaTeX-Dokuments die folgende Zeile einfügen (das German-Paket auch gleich noch mit):

```
\usepackage[latin1]{inputenc}
\usepackage{ngerman}
```

Wenn Sie die LaTeX-Datei als Unicode-Datei (UTF-8) speichern, müssen Sie am Beginn des LaTeX-Dokuments die folgende Zeile einfügen. Bei manchen Distributionen funktioniert das allerdings erst nach der Installation eines Extrapakets (unter Fedora `tetex-unicode`).

```
\usepackage[utf8x]{inputenc}
```

Das Unicode-Paket muss oft eigens installiert werden. Wenn das Paket bei Ihrer Distribution fehlt, finden Sie es im Internet unter *http://www.unruh.de/DniQ/latex/unicode/*. Zukünftige LaTeX-Versionen werden eine UTF-8-Encoding-Datei direkt enthalten. Die Notwendigkeit des UCS-Pakets entfällt dann.

8.2.3 Maßangaben

Zahlenwerte in Kommandos erfolgen bei allen Maßangaben als Dezimalzahl. Sie können sowohl mit einem Dezimalpunkt als auch mit einem Komma angegeben werden. Der Zahlenwert muss immer mit einer Einheit abgeschlossen werden (auch bei 0). Die Angabe kann nicht nur absolut, sondern auch als Faktor einer bekannten Größe (z. B. `0.6 \textwidth` erfolgen. Als Einheiten sind folgende Werte möglich:

Maßeinheiten

sp	1 scaled point = 1/65536 pt
pt	1 point = 1/72,269 Zoll = 0,351 mm
bp	1 big point = 1/72 Zoll
dd	1 Didôt point = 1/72 französische Zoll = 0,376 mm
mm	1 Millimeter = 2,845 pt
pc	1 pica = 12 pt = 4,218 mm
cc	1 Cicero = 12 dd = 4,531 mm
cm	1 Centimeter = 2,371 pc
in	1 inch = 72,269 pt = 25,4 mm = 6,022 pc
ex	die Höhe eines x in der aktuellen Schriftart
em	die Breite eines Gedankenstrichs (–) in der aktuellen Schriftart

8.2.4 Strukturierung von Texten

Der eigentliche Text wird mit dem Kommando `\begin{document}` eingeleitet und mit `\end{document}` abgeschlossen. Der Text innerhalb dieser beiden Kommandos wird dann durch LATEX verarbeitet und gesetzt. Dabei stehen einige Kommandos zur Verfügung, mit denen der Text strukturiert werden kann:

Dokumentstrukturierung	
`\part{Überschrift}`	Teil (nur für `book` und *report*)
`\chapter{Überschrift}`	Kapitel (nur für `book` und *report*)
`\section{Überschrift}`	Abschnitt
`\subsection{Überschrift}`	Unterabschnitt
`\subsubsection{Überschrift}`	Unterunterabschnitt

LATEX kümmert sich selbstständig um die Nummerierung der Kapitel und Abschnitte. Gleichzeitig wählt LATEX automatisch geeignete Schriftarten und Textabstände für die Überschriften. Beim Texttyp *book* werden die Texte des `\chapter`- und `\section`-Kommandos auch beim Erstellen der Kopfzeilen berücksichtigt. Wenn die Kommandos in der Form `\section[kurzfassung]{vollständig}` verwendet werden, wird die Kurzfassung für die Kopfzeile und das Inhaltsverzeichnis, die vollständige Variante dagegen unmittelbar im Text verwendet.

Falls der Text mit einem Anhang ausgestattet werden soll, wird dieser mit `\appendix` eingeleitet. Innerhalb des Anhangs können wieder `\chapter`, `\section` etc. verwendet werden, wobei Kapitel jetzt mit A, B, C ... nummeriert werden.

Ein typisches LATEX-Dokument beginnt meist so ähnlich wie das folgende Beispiel:

```
\documentclass[a4paper,11pt]{article}  % Dokumententyp
\usepackage{ngerman}                   % Zusatzpakete
\usepackage[latin1]{inputenc}
```

```
% eventuell Inhaltsverzeichnis einfügen
\tableofcontents

%Anzahl der Warnungen (underfull/overfull) reduzieren (es bleiben genug!)
\sloppy

\begin{document}
% der eigentliche Text, strukturiert durch
% \section{...}, \subsection{...} etc.

\appendix
% der Text des Anhangs, wiederum strukturiert durch
% \section{...}, \subsection{...} etc.
\end{document}
```

8.2.5 Gestaltung des Schriftbilds

Von herkömmlichen Textverarbeitungsprogrammen sind Sie vermutlich gewohnt, dass Sie bei der Auswahl von Schriftarten und -größen praktisch unbeschränkt flexibel sind. In LaTeX ist das aus historischen Gründen nicht der Fall. Im Regelfall sind Sie in LaTeX auf drei Schriftfamilien beschränkt: eine Standardschrift, die für den Fließtext, die Überschriften etc. verwendet wird, die Schrift `Typewriter` für Programmlistings und die Schrift Sans Serif für Hervorhebungen und andere Aufgaben.

Diese drei Schriftfamilien können in unterschiedlichen Attributen formatiert werden. Die meisten Attribute existieren bei der Standardschrift, die fett, kursiv, geneigt und in Kapitälchen dargestellt werden kann.

Die Unterscheidung zwischen kursiv und geneigt wird Ihnen vermutlich unbekannt vorkommen: Bei geneigten Schriften verwendet LaTeX die normale Schrift und neigt die Buchstaben ein wenig nach rechts. Bei kursiven Schriften wird dagegen eine eigene Schriftart verwendet, in der die Zeichen stärker geneigt und auch ein wenig anders geformt sind. Diese Unterscheidung gilt allerdings nicht bei allen Schriftarten.

Standardschrift	normal	*kursiv*	*geneigt*	KAPITÄLCHEN
	fett	***fett kursiv***	***fett geneigt***	**fett Kapitälchen**
Sans Serif	normal	*kursiv*	*geneigt*	KAPITÄLCHEN
	fett	**fett kursiv**	**fett geneigt**	**fett Kapitälchen**
`Typewriter`	`normal`	*`kursiv`*	*`geneigt`*	`KAPITÄLCHEN`

Aus historischen Gründen gibt es mehrere alternative Kommandos zur Einstellung der Schriftfamilien und -attribute. In der folgenden Tabelle sind in den

beiden ersten Spalten die offiziellen Kommandos von LaTeX 2ε, in der dritten Spalte die noch immer erlaubte alte Syntax und in der vierten Spalte das Resultat aufgelistet. Das Ergebnis der verschiedenen Kommandos ist nur scheinbar dasselbe (siehe die folgende Tabelle).

`\emph` schaltet nicht einfach auf kursive Schrift um, sondern wechselt das Schriftattribut zwischen normal und kursiv. Innerhalb einer kursiven Schrift liefert `\emph` daher eine normale Schrift.

Die Kurzkommandos wie `\sl`, `\bf` etc. wirken zwar wegen des minimalen Tippaufwands attraktiv, haben aber aus Kompatibilitätsgründen eine recht eigenwillige Funktionsweise: So wechseln `\bf`, `\it`, `\sl` und `\it` in die Standardschriftart und deaktivieren alle anderen Attribute! Es ist mit diesen Kommandos also nicht möglich, Schriftattribute zu kombinieren. Verwenden Sie unbedingt die neueren `\textxy`-Kommandos, etwa `\textbf{\textit{text}}`!

Schriftattribute

LaTeX 2ε		**LaTeX2.09**	**Ergebnis**
`\textrm{text}`	`{\rmfamily text}`	`{\rm text}`	Standard (Roman)
`\textsf{text}`	`{\sffamily text}`	`{\sf text}`	Sans Serif
`\texttt{text}`	`{\ttfamily text}`	`{\tt text}`	`Typewriter`
`\textbf{text}`	`{\bferies text}`	`{\bf text}`	**fett**
`\textmd{text}`	`{\mdseries text}`	`{\rm text}`	normal
`\textit{text}`	`{\itshape text}`	`{\it text}`	*kursiv (italic)*
`\textsl{text}`	`{\slshape text}`	`{\sl text}`	*geneigt (slanted)*
`\textsc{text}`	`{\itshape text}`	`{\it text}`	*Kapitälchen*
`\textup{text}`	`{\upshape text}`	`{\rm text}`	normal
`\emph{text}`		`{\em text}`	*hervorgehoben*

Auch bei den Schriftgrößen unterscheidet sich LaTeX von dem, was Sie von anderen Textverarbeitungsprogrammen gewohnt sind. Es gibt eine Standardschriftgröße, die für das gesamte Dokument gilt. Diese Schriftgröße kann nur zwischen 10 und 12 Punkt variieren. 10 Punkt ist die Default-Einstellung; auf 11 oder 12 Punkt können Sie durch die `\documentstyle`-Optionen `11pt` oder `12pt` umstellen.

Von dieser Standardschriftgröße ausgehend, berechnet LaTeX automatisch passende Schriftgrößen für Überschriften durch `\chapter` oder `\section`, für Fußnoten, mathematische Formeln etc. Gleichzeitig gilt diese Größe als Anhaltspunkt für die Kommandos zur Veränderung der Schriftgröße:

Schriftgröße

`\tiny`	nur noch mit der Lupe zu lesen	`\Large`	größer
`\scriptsize`	winzig	`\LARGE`	noch größer
`\footnotesize`	sehr klein		
`\small`	klein	`\huge`	riesig
`\normalsize`	Standardschrift		
`\large`	groß	`\Huge`	kingsize

Um Text hoch[zustellen], verwenden Sie `\textsuperscript{text}`. In mathematischen Formeln gilt dagegen die Schreibweise `$a^{b}$` für a^b bzw. `$a_{10}$` für a_{10}.

8.2.6 Sonderzeichen

Sehr viele Sonderzeichen wie `%` oder `$` gelten in LaTeX als Kommandos. Einige weitere Sonderzeichen haben zwar nur im mathematischen Modus eine besondere Bedeutung (z. B. `^` und `_`), können aber im normalen Textmodus ebenfalls nicht verwendet werden. Die folgende Tabelle fasst die Bedeutung der wichtigsten Sonderzeichen zusammen:

Bedeutung von Sonderzeichen	
`%`	leitet Kommentare ein
`~`	festes Leerzeichen (z. B. in `5~cm`)
`{..}`	klammert Textbereiche ein (z. B. für besondere Formatierung)
`$formel$`	klammert Formeln im Fließtext ein
`_`	tiefstellen (nur im mathematischen Modus)
`^`	hochstellen (nur im mathematischen Modus)

Der Versuch, Sonderzeichen im laufenden Text unverändert darzustellen, ist eine Quelle beständigen Ärgers: Zum einen wird es Ihnen auch nach monatelangem Arbeiten mit LaTeX noch passieren, dass Sie einfach übersehen, dass ein Zeichen eine besondere Bedeutung hat. Zum anderen fehlt eine einheitliche Methode, um Sonderzeichen im Text darzustellen. In vielen Fällen reicht es, wenn dem Sonderzeichen einfach ein Backslash vorangestellt wird (etwa `\%`, um ein %-Zeichen zu erzeugen). Wenn das nichts hilft, kann der Code des Zeichens direkt mit `\char`*n* angegeben werden. Bei den meisten Textzeichen gilt dabei der normale ASCII-Code.

Die folgende Tabelle fasst die Kommandos zur Darstellung der wichtigsten Sonderzeichen zusammen. Beachten Sie, dass es bei manchen Zeichen unterschiedliche Varianten gibt, die sich typografisch oft ein wenig unterscheiden.

Darstellung von Sonderzeichen

<table>
<tr><td>--</td><td>– (Gedankenstrich)</td><td>\char34{}</td><td>”</td></tr>
<tr><td>---</td><td>— (langer Strich)</td><td>\textbackslash</td><td>\</td></tr>
<tr><td>_</td><td>_ (Unterstrich)</td><td>{\tt<}</td><td><</td></tr>
<tr><td>{\tt\char95}</td><td>_ (Unterstrich)</td><td>{\tt>}</td><td>></td></tr>
<tr><td>\#</td><td>#</td><td>\textless</td><td><</td></tr>
<tr><td>\$</td><td>$</td><td>\textgreater</td><td>></td></tr>
<tr><td>\&</td><td>&</td><td>\textasciitilde</td><td>˜</td></tr>
<tr><td>\%</td><td>%</td><td>\pounds</td><td>£</td></tr>
<tr><td>\{</td><td>{</td><td>\copyright</td><td>©</td></tr>
<tr><td>\}</td><td>}</td><td>"‘ („xxx...)</td><td>„</td></tr>
<tr><td>\textbar</td><td>|</td><td>"’ (...xxx“)</td><td>“</td></tr>
<tr><td>\textasciicircum</td><td>ˆ</td><td></td><td></td></tr>
</table>

Bei der Anwendung der obigen Kommandos ist Vorsicht geboten: LaTeX eliminiert nach manchen Kommandos ein eventuell erwünschtes Leerzeichen. So wird aus \copyright Michael Kofler „©Michael Kofler“. Damit zwischen © und dem folgenden Text ein Leerzeichen angezeigt wird, muss das Kommando mit einem Backslash oder mit geschwungenen Klammern abgeschlossen werden, also \copyright\ Michael Kofler oder \copyright{} Michael Kofler.

Zahlreiche Sonderzeichen können sehr einfach im mathematischen Modus gebildet werden – etwa π mit `$\pi$`, $\rightarrow$ mit `$\rightarrow$`, $<$ mit `$<$` oder $\leftarrow$ mit `$\leftarrow$`. Beachten Sie aber, dass dabei eine andere Schrift verwendet wird. Eine Zusammenstellung der wichtigsten mathematischen Zeichen finden Sie ab Seite 387.

Einen alternativen Weg zur Darstellung von Texten mit Sonderzeichen bietet das Kommando \verb_text_. \verb_ls | more_ liefert ls | more. In diesem Beispiel wurde der Unterstrich als Klammerzeichen für den eigentlichen Text verwendet. Sie können aber ebenso ein beliebiges anderes Zeichen verwenden, das im darzustellenden Text nicht vorkommt. Der Text innerhalb von \verb wird in der Schriftart typewriter dargestellt. \verb kann in zahlreichen Umgebungen und insbesondere zur Definition von Makros nicht verwendet werden. \verb-Texte sind auf maximal eine Zeile beschränkt. Es gibt keine Möglichkeit, \verb-Texte fett darzustellen.

Wenn Sie nicht nur einige Zeichen oder Wörter mit Sonderzeichen darstellen möchten, sondern mehrere Zeilen, bietet sich dazu die verbatim-Umgebung an. Diese Umgebung wird mit \begin{verbatim} eingeleitet und mit \end{verbatim} abgeschlossen. Alle dazwischen angegebenen Programmzeilen werden unverändert in der typewriter-Schriftart dargestellt. Die verbatim-Umgebung eignet sich insbesondere zur Wiedergabe von Programmlistings, in denen es zumeist von Sonderzeichen nur so wimmelt.

verbatim-Umgebung

`\verb_text_`	stellt den Text inklusive aller Sonderzeichen dar
`\begin{verbatim}`	leitet Text ein, der unverändert ausgegeben wird
`Beispieltext mit`	
`Sonderzeichen $ % ^ ~`	
`\end{verbatim}`	Ende der `verbatim`-Umgebung

Statt der `verbatim`-Umgebung können Sie auch die `alltt`-Umgebung einsetzen, wenn Sie das gleichnamige Zusatzpaket aktiviert haben. Damit können Sie innerhalb der Umgebung die Schriftart durch LATEX-Kommandos verändern. Allerdings werden die Zeichen {, }, |und \ nun als LATEX-Zeichen interpretiert.

8.2.7 Akzente und besondere Buchstaben

Die meisten westeuropäischen Sonderzeichen (Latin-1) können direkt im Text angegeben werden, sofern das LATEX-Dokument einen geeigneten Zeichensatz verwendet (z. B. `\usepackage[latin1]{inputenc}`, siehe Seite 349). Darüber hinaus bietet LATEX aber auch die Möglichkeit, Buchstaben und Akzente beinahe beliebig zu kombinieren. Die folgende Tabelle zeigt die wichtigsten Kombinationsmöglichkeiten für den Buchstaben a.

Buchstaben und Akzente kombinieren							
`\'a`	á	`` \`a ``	à	`\^a`	â	`\~a`	ã
`\.a`	ȧ	`\=a`	ā	`\b{a}`	a̲	`\d{a}`	ạ
`\u{a}`	ă	`\v{a}`	ǎ	`\c{a}`	a̧	`\"{a}`	ä

Für einige weitere Buchstaben gibt es besondere Codes: `\ae`, `\AE`, `\aa`, `\AA`, `\oe`, `\OE`, `\o` und `\O` liefern æ, Æ, å, Å, œ, Œ, ø und Ø.

8.2.8 Euro-Symbol

LATEX selbst kennt zwar kein Euro-Symbol, es gibt aber eine Reihe von Möglichkeiten, das Zeichen in LATEX-Dokumenten einzufügen. Das Paket `textcomp` stellt ein Euro-Symbol mit dem Kommando `\texteuro` zur Verfügung: €

Das Paket `eurosym` stellt ein Euro-Symbol mit dem Kommando `\euro` zur Verfügung und zwar auch fett und kursiv. Eine weitere Variante besteht darin, selbst ein Euro-Symbol zu definieren, indem Sie die folgenden Zeilen an den Beginn des LATEX-Dokuments stellen. `\myeuro` liefert nun das Symbol €, das allerdings nicht exakt dem offiziellen Symbol entspricht (Haben Sie es bemerkt? Unser erstes eigenes Makro!).

```
\newcommand\myeuro{{\sffamily C%
  \makebox[0pt][l]{\kern-.70em\mbox{--}}%
  \makebox[0pt][l]{\kern-.68em\raisebox{.25ex}{--}}}}
```

8.2.9 Tabulatoren

Mit LaTeX haben Sie mehrere Möglichkeiten zur Definition von Tabellen. Die einfachste Variante ist die`tabbing`-Umgebung. Die Syntax dieser Umgebung sieht folgendermaßen aus:

Tabellen

`\begin{tabbing}`	
`muster \= muster \= \kill`	Musterzeile: `\=` definiert Tabulatoren
`term1 \> term2 \> term3\\`	Tabelle: `\>` für Tabulator, `\\` für Zeilenende
`term4 \> term5`	letzte Zeile ohne `\\`
`\end{tabbing}`	

Die Tabellen werden also mit einer Musterzeile eingeleitet. In dieser Musterzeile werden die Positionen der linksbündigen Tabulatoren mit `\=` festgelegt. Als Musterzeile verwenden Sie normalerweise die breiteste Zeile aus Ihrer Tabelle. In diese Zeile sollten Sie vor jedem `\=` einen zusätzlichen Leerraum einfügen, beispielsweise mit einer `\qquad`-Anweisung (Leerraum in der Größe zweier Gedankenstriche (siehe Seite 390). `\kill` löscht die Musterzeile, sodass diese Zeile nur als Muster verwendet, aber nicht ausgegeben wird. Dazu ein Beispiel: Die Sonderzeichentabelle von Seite 353 wurde mit den folgenden LaTeX-Anweisungen produziert:

```
\begin{tabbing}
{\verb?$formel$?}\qquad\=\kill
{\verb?%?}
  \> leitet Kommentare ein\\
{\verb?~?}
  \> festes Leerzeichen (z.\,B.\ in {\verb?5~cm?})\\
{\verb?{..}?}
  \> klammert Textbereiche ein (z.\,B.\ für eine besondere
     Formatierung)\\
{\verb?$formel$?}
  \> klammert Formeln im Fließtext ein\\
{\verb?_?}
  \> tiefstellen (nur im mathematischen Modus)\\
{\verb?^?}
  \> hochstellen (nur im mathematischen Modus)
\end{tabbing}
```

Wenn Ihnen diese einfache Form von Tabellen nicht ausreicht, bietet LaTeX die `tabular`-Umgebung an: Damit können Sie Tabellen mit wechselnder Spaltenbreite, mit Umrandung etc. erzeugen. Wenn Sie außerdem noch die `table`-Umgebung einsetzen, wird die Tabelle je nach Platzangebot automatisch wie ein Bild platziert, ohne dass Löcher im Text entstehen.

8.2.10 Tabellen

Die wichtigsten Möglichkeiten zum Setzen von Tabellen in LaTeX bieten die Umgebungen `tabular`, `tabular*` und `array`. Diese drei Umgebungen sind bereits in den Standarddokumentenklassen enthalten, wobei die `array`-Umgebung nur im mathematischen Modus verwendet werden kann. Die Syntax und die Bedeutung der Parameter aller drei Umgebungen stimmen überein.

Tabellen können beliebig ineinander verschachtelt werden, man kann sogar innerhalb einer Zelle einer Tabelle wieder eine Tabelle einbauen, es sollte jedoch jede Einzeltabelle innerhalb einer Gruppe eingeschlossen sein, was bei `tabularx`-Tabellen sogar zwingend erforderlich ist.

Tabellen werden von LATEX als untrennbare Einheit betrachtet, sie müssen deshalb immer auf die Seite passen. Passt eine Tabelle nicht mehr auf eine Seite, wird sie auf die folgende Seite gesetzt. Es gibt aber Pakete, die mehrseitige Tabellen ermöglichen.

Tabellen bestehen aus der Tabellenpräambel, in welcher der Typ der Tabelle und die Formatierung der Spalten deklariert werden, und dem Tabellenkörper, der von den Einträgen gebildet wird. Für die `tabular`-Umgebungen ergibt sich somit folgender Aufbau:

```
% Tabellenpräambel:
\begin{tabular}[position]{spaltenformat}
% Tabellenkörper:
Einträge
\end{tabular}
```

Alle Angaben in der Tabellenpräambel gelten global für die gesamte Tabelle, es gibt jedoch Befehle, mit denen man im Tabellenkörper die globale Festlegung für einzelne Zellen oder Zeilen ändern kann. Die drei Umgebungen haben unterschiedliche Eigenschaften:

- `tabular` für „normale“ Tabellen. Die Breite hängt von den Einträgen und den Abständen zwischen den Spalten ab, LATEX richtet die Breite der einzelnen Spalten nach den Zelleneinträgen aus. Man muss nur darauf achten, dass die Tabelle nicht zu breit wird.
- `tabular*` hat einen zusätzlichen Parameter, der explizit die Gesamtbreite der Tabelle vorgibt:
  (`\begin{tabular*}{breite}[position]{spaltenformat}`
- `array` dient der Darstellung von Matrizen, Tensoren usw. und kann nur im mathematischen Modus verwendet werden.

Die Ausrichtung einer Tabelle erfolgt auf die laufende Zeile des Textes und wird immer durch den Positions-Parameter festgelegt. Diesr Parameter ist optional; wird er nicht angegeben, erfolgt eine zentrierte Ausrichtung.

t Die Tabelle wird so in den Text eingefügt, dass die oberste Tabellenzeile bündig mit dem laufenden Text abschließt.

c Die Tabelle wird so in den Text eingefügt, dass sie zentriert zum laufenden Text steht.

b Die Tabelle wird so in den Text eingefügt, dass die unterste Tabellenzeile bündig mit dem laufenden Text abschließt.

Wird die Tabelle als Gleitobjekt (Umgebung `table`, siehe unten) gesetzt, kann der Parameter `position` in der Tabellenpräambel weggelassen werden, da die `table`-Umgebung eine höhere Priorität besitzt und somit die Ausrichtung bestimmt.

Mit dem Parameter `spaltenformat` werden die Anzahl der Spalten, die Ausrichtung der Einträge sowie die Rahmen- und Zwischenlinien der Tabelle definiert. Mit weiteren Befehlen kann gegebenenfalls auch die Breite der Spalten oder ein Text zwischen zwei Spalten festgelegt werden. Die einfachen Formatierungsoptionen sind:

`|`: vertikale Linie über die gesamte Tabellenhöhe
`l`: linksbündige Einträge
`c`: zentrierte Einträge
`r`: rechtsbündige Einträge

Besondere Optionen für das Spaltenformat sind:

- **p{breite}**: Eine so genannte `parbox`-Spalte der Breite `breite`, wobei der Eintrag am oberen Rand der Box ausgerichtet wird. Damit lassen sich mehrzeilige Texte in einer Spalte unterbringen (siehe unten).
- ***{anzahl}{format}**: Abkürzende Schreibweise für ein Format, dabei treten die in `format` angegebenen Optionen `anzahl`-mal auf
- **@{text}**: Diese Option fügt den angegebenen Text oder Befehl an der Stelle in die Tabelle ein, an der die Option in der Präambel steht.

Wird `p{breite}` als Option verwendet, steht der Eintrag innerhalb einer Box, deren Rahmen nicht sichtbar ist. Die Ausrichtung des Textes erfolgt am oberen, nicht sichtbaren Rand der Box. Dieser Befehl wird verwendet, wenn man die Breite der einzelnen Spalten explizit vorgeben will oder ein längerer Text mehrzeilig in der Spalte stehen soll. LaTeX umbricht die Zeilen der Box in Abhängigkeit von der vorgegebenen Breite, es ist aber auch möglich, den Zeilenumbruch mit den Befehlen `\newline` oder `\linebreak` vorzugeben (nicht jedoch mit `\\` oder `\tabularnewline` versuchen, da dies eine neue Zeile in der Tabelle ergibt).

Bei der Verwendung der Option `@{text}` wird der vorgegebene Abstand zwischen den beiden Spalten auf 0 gesetzt und stattdessen der Text eingefügt. Will man den Spaltenabstand unterdrücken, kann man das mittels `@{}` erreichen.

Im Tabellenkörper werden die eigentlichen Daten der Tabelle angegeben und die horizontalen Linien der Tabelle formatiert. Die Eintragung der Daten in die Tabelle erfolgt zeilenweise, einzelne Spalteneinträge werden durch ein `&` voneinander getrennt. Die Anzahl der Spalten muss mit der in der Tabellenpräambel definierten Spaltenzahl übereinstimmen. Die Tabellenzeilen werden durch den Befehl `\\` oder `\tabularnewline` abgeschlossen (Pflicht!).

Nun wird es aber Zeit für ein paar Beispiele:

```
\begin{tabular}{lcr}
{\bf Links} & {\bf Mitte} & {\bf Rechts} \\
111 & 222 & 333 \\
Aus & die & Maus \\
\end{tabular}
```

Ergibt:

Links	**Mitte**	**Rechts**
111	222	333
Aus	die	Maus

Mit senkrechten Strichen sieht das dann folgendermaßen aus:

```
\begin{tabular}{l|c|r}
{\bf Links} & {\bf Mitte} & {\bf Rechts} \\
111 & 222 & 333 \\
Aus & die & Maus \\
\end{tabular}
```

Führt zu:

Links	**Mitte**	**Rechts**
111	222	333
Aus	die	Maus

Hat die Tabelle längere Einträge, gehen bei den Spaltenformaten „r“, „c“ und „l“ die Zeilen irgendwann über den Rand hinaus. Da hilft dann nur noch das Format „p“, wie in folgendem Beispiel (Tabelle 8.1, das auch gleich noch weitere Informationen über die Gestaltung von Tabellen liefert. Hier setzen wir auch die vertikale und horizontale Spaltenbegrenzung ein (im Listing wurde der Text gekürzt):

```
\begin{tabular}{|p{0.45 \textwidth}|p{0.45 \textwidth}|}
\hline
\verb?\hline? & setzt eine horizontale Linie über die ... \\
\hline
\verb?\cline{s1-s2}? & setzt ein horizontales Linienstück ... \\
```

Tabelle 8.1: Tabelle mit zwei Spalten, die längeren Text enthalten

`\hline`	setzt eine horizontale Linie über die gesamte Breite der Tabelle. Dieser Befehl darf nur am Anfang einer Tabellenzeile verwendet werden.
`\cline{s1-s2}`	setzt ein horizontales Linienstück von Spalte *s1* bis Spalte *s2* der Tabelle. Dieser Befehl darf nur am Anfang einer Tabellenzeile verwendet werden.
`\vline`	setzt eine vertikale Linie über die Höhe der Zeile. `\vline` wird verwendet, wenn nicht die gesamte Tabelle eine vertikale Linie erhalten soll.
`\multicolumn{anz}{form}{text}`	macht aus den nächsten `anz` Spalten eine Spalte von der Gesamtbreite der `anz` Spalten einschließlich Zwischenräumen. Erlaubt sind die Format-Optionen `l`, `c` und `r` sowie ggf. senkrechte Striche davor oder danach, ebenso der `@{text}`-Befehl.

```
\hline
\verb?\vline? & setzt eine vertikale Linie über die Höhe ... \\
\hline
\verb?\multicolumn{anz}{form}{text}? & macht aus den nächsten ... \\
\hline
\end{tabular}
```

Der Befehl `cline{s1-s2}` darf auch mehrmals nacheinander auftreten (ebenso `\hline`). Soll die horizontale Linie sich nur über eine Spalte erstrecken, sind `s1` und `s2` gleich – es müssen immer zwei Werte angegeben werden.

Einen anders ausgerichteten Text in einer Zelle gegenüber der restlichen Tabelle erhält man, wenn man den Befehl `\multicolumn{1}{format}{text}` verwendet – also nur für eine Spalte. Da der Befehl die in der Tabellenpräambel vorgegebene Formatierung überdeckt, müssen ggf. senkrechte Striche mit angegeben werden.

```
\begin{tabular}{|l|l||l|}
\hline
{\bf System} & {\bf Multiuser} & {\bf Multitasking} \\
\hline
\hline
MS-DOS & Nein & Nein \\
```

```
\cline{1-2}
CP/M & Nein &  \\
\hline
Windows & Nein & Ja \\
\cline{1-2}
Linux & Ja &  \\
\hline
\end{tabular}
```

Ergibt:

System	**Multiuser**	**Multitasking**
MS-DOS	Nein	Nein
CP/M	Nein	
Windows	Nein	Ja
Linux	Ja	

Vorsicht ist bei schmalen Tabellen geboten, da es zu Problemen mit der Worttrennung kommen kann. LaTeX trennt das erste Wort normalerweise nicht. Bei sehr schmalen Spalten kann das aber erforderlich sein (wenn es sich beispielsweise sowieso nur um ein langes Wort handelt). In solchen Fällen hilft das Einfügen eines horizontalen Freiraums der Länge 0 vor dem Wort (`\hspace{0 pt}`). Der wird von LaTeX wie ein Wort betrachtet – und das zweite Wort kann getrennt werden. Dazu ein Beispiel (man beachte das gruselige „fff“):

```
\begin{tabular}{|p{1.5cm}|p{3.0cm}|}
\hline
Donaudampfschifffahrtsgesellschaftskapitän &
Donaudampfschifffahrtsgesellschaftskapitän \\
\hline
\end{tabular}
```

Ergibt:

Donaudampfschifffahrtsgesellschaftskapitän	Donaudampfschifffahrtsgesellschaftskapitän

Dagegen erreicht man mit:

```
\begin{tabular}{|p{1.5cm}|p{3.0cm}|}
\hline
\hspace{0pt}Donaudampfschifffahrtsgesellschaftskapitän &
\hspace{0pt}Donaudampfschifffahrtsgesellschaftskapitän \\
\hline
\end{tabular}
```

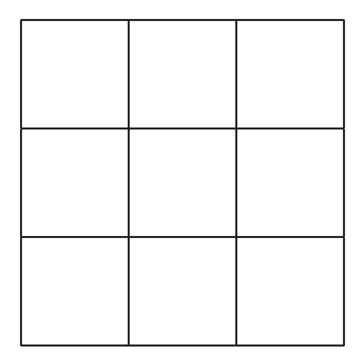
Das gewünschte Ergebnis:

Donau- dampf- schiff- fahrts- gesell- schaftska- pitän	Donaudampfschiff- fahrtsgesellschafts- kapitän

Innerhalb eines jeden Feldes der Tabelle sind wieder alle Formatierungsangaben sowie Veränderungen der Schriftgröße und -art möglich.

Aber nicht nur die Breite der Spalten kann beeinflusst werden. Mithilfe des Befehls `\vspace{Hoehe}` kann auch die Höhe einer Tabellenzeile angepasst werden und Sie können so die Größe der Tabellenfelder in beiden Dimensionen festlegen, z. B.:

```
\begin{tabular}{|p{1cm}|p{1cm}|p{1cm}|}
\hline
\vspace{1cm} & \vspace{1cm} & \vspace{1cm} \\
\hline
\vspace{1cm} & \vspace{1cm} & \vspace{1cm} \\
\hline
\vspace{1cm} & \vspace{1cm} & \vspace{1cm} \\
\hline
\end{tabular}
```



Schwierig wird es für den Anfänger, wenn Zahlen formatiert werden sollen. Bei der folgenden Tabelle gibt es zwei Spalten. In der ersten Spalte steht eine Mengenangabe, in der zweiten eine Bezeichnung. Die Tabelle ist als solche im Text gar nicht zu erkennen. Im Gegensatz zur `\tabbing`-Umgebung haben Sie aber die Möglichkeit der rechts- bzw. linksbündigen Positionierung:

```
\begin{tabular}{rl}
100 & RAMs 512 MByte, PC 133 \\
52 & RAMs 256 MByte, PC 133 \\
10 & PS2-RAMS 64 MByte \\
350 & Festplatten 80 GByte\\
```

```
\end{tabular}
```

Stellt sich im Text dar als:

100	RAMs 512 MByte, PC 133
52	RAMs 256 MByte, PC 133
10	PS2-RAMS 64 MByte
350	Festplatten 80 GByte

Was aber, wenn auch noch Dezimalstellen ins Spiel kommen, etwa bei einer Preisliste. Kein Problem, wenn Sie den Trick kennen. Bei den Preisen werden einfach die Stellen vor dem Komma und jene dahinter in zwei Tabellenspalten untergebracht. Der Dezimalpunkt oder das Dezimalkomma dienen dann als Spaltentrenner:

```
\begin{tabular}{|l|r@{,}l@{ Euro }|}
\hline
RAMs 512 MByte, PC 133 & 45 & 50 \\
\hline
RAMs 256 MByte, PC 133 & 28 & 99 \\
\hline
PS2-RAMS 64 MByte & 9 & 95 \\
\hline
Festplatten 80 GByte & 121 & 60 \\
\hline
\end{tabular}
```

Ratz-Fatz ist eine ordentliche Preisliste fertig. Mit ein paar Zeilen Perl könnte man den LaTeX-Quelltext sogar aus den Daten einer Kalkulationstabelle oder den Ergebnissen einer Datenbankabfrage vollautomatisch generieren. Für den speziellen Fall der Zahlenformatierung gibt es natürlich ein passendes Makropaket – was dem Beispiel aber keinen Abbruch tut.

RAMs 512 MByte, PC 133	45,50 Euro
RAMs 256 MByte, PC 133	28,99 Euro
PS2-RAMS 64 MByte	9,95 Euro
Festplatten 80 GByte	121,60 Euro

8.2.11 Gleitobjekte

Leider gibt es bei solchen Tabellen, aber auch bei Bildern, ein kleines Problem. Die Tabelle wird als zusammenhängender Teil des Satzsystems betrachtet – im Grunde wie ein einzelner Buchstabe. Passt sie nicht mehr komplett auf die Seite, wird sie auf die folgende Seite gesetzt.

Beim Gestalten von Artikeln und Büchern gilt die Regel, dass Tabellen, Bilder usw. auch etwas verschoben gesetzt werden dürfen – etwa auf benachbarte Seiten. Der Text läuft dann beispielsweise bis zum Ende der Seite weiter und

auf der folgenden Seite erscheint dann die Tabelle. In diesem Fall kann man Tabellen, Bilder usw. auch mit einer Legende (Bildunterschrift) und Nummerierung versehen und sich im Text darauf beziehen („... weitere Optionen sind in Tabelle 3.6 aufgeführt ...“).

Solche Textelemente nennt man in LATEX „Gleitobjekte“, weil sie eben im Text verschieblich (gleitend) angeordnet sind. Manchmal machen solche Gleitobjekte Probleme bei der Anordnung. Kann ein Objekt nicht platziert werden, schiebt das LATEX-System das Objekt ans Kapitelende (oder bis zum nächsten `\newpage`). Alle folgenden Gleitobjekte werde dahintergepackt, weil ja die Reihenfolge erhalten bleiben muss, und schon stehen alle Tabellen ganz hinten. Da hilft dann nur Trixen, z. B. die Tabelle etwas weiter nach vorne setzen oder in zwei Abteilungen aufteilen. Auch ein `\newpage`-Befehl an der richtigen Stelle hilft. Dass das Problem nicht trivial ist, zeigen etliche Dokumente zur Platzierung von Gleitobjekten in der LATEX-Dokumentation.

Für Tabellen steht die Gleitobjektumgebung `table` zur Verfügung, deren Grundkonzept kurz vorgestellt werden soll. Das Grundgerüst der `table`-Umgebung ist wie jede andere Umgebung in LATEX aufgebaut:

```
\begin{table}[ausrichtung]
{Inhalt}
\end{table}
```

Möchte man in einem zweispaltiges Layout eine Tabelle über die gesamte Seitenbreite setzen, so ist die `table*`-Umgebung zu verwenden, die im Übrigen die gleichen Eigenschaften besitzt.

Für die Positionierung der Umgebung dient der Parameter `ausrichtung`, welcher optional verwendet wird. Für diesen Parameter stehen fünf Angaben zur Verfügung:

h LATEX setzt das Gleitobjekt an der Stelle, wo es im Quelltext vorkommt, wenn dies möglich ist, andernfalls setzt LATEX das Gleitobjekt an den Seitenanfang. Bei zwei aufeinanderfolgenden „h“-Gleitobjekten ist es besser, diese in eine einzige Gleitobjektumgebung zu setzen.

t Das Gleitobjekt wird am oberen Seitenrand gesetzt.

b Das Gleitobjekt wird am unteren Seitenrand gesetzt.

p Das Gleitobjekt wird auf einer Gleitobjektseite gesetzt, alle folgenden Gleitobjekte können erst nach Ausgabe dieser Seite gesetzt werden.

! hinter h, t oder b bewirkt, dass LATEX seine Voreinstellungen weitgehend außer Acht lässt, um die vorgeschriebene Positionierung vorzunehmen

Es können mehrere Parameter verwendet werden, wobei die Auswertung nach der angegebenen Reihenfolge erfolgt. Die Verwendung der Parameter ist optional.

Außerdem besteht die Möglichkeit, dass Gleitobjekt, in unserem Fall die Tabelle, durch den Befehl `\caption{text}` mit einem Text zu beschriften und mit einem Label zu versehen (`\label{labeltext}`). Die Tabellen werden dann kapitelweise in der Form „Kapitelnummer.Tabellennummer“ nummeriert. Mit dem Kommando `\ref{labeltext}` können Sie sich im Fließtext dann auf die Nummer der Tabelle beziehen und mit dem Befehl `\listoftables` lässt sich ein Tabellenverzeichnis erstellen.

Die Tabelle kann mit dem Befehl `\caption` in der Gleitobjektumgebung mit einer Legende versehen werden. Die eigentliche Tabelle steht innerhalb der `table`-Umgebung. Insgesamt sieht das dann aus wie im folgenden Listing, wo die Tabelle auch gleich noch zentriert wird:

```
\begin{table}[h!t]
\begin{center}
\caption{Tabellenüberschrift \label{tra-la-la}}
\begin{tabular}
...
\end{tabular}
\end{center}
\end{table}
```

8.2.12 Aufzählungen

In LaTeX sind mehrere Möglichkeiten zur Formatierung von Aufzählungen vorgesehen. Die einfachste Variante stellt die `itemize`-Umgebung dar. Die `itemize`-Umgebung wird zur einfachen Eingabe von Aufzählungen, Listen oder Ähnlichem benutzt; es ist eine Verschachtelung bis zur Tiefe 4 möglich, mit jeder Verschachtelung wird etwas mehr nach rechts eingerückt. Als Marken werden standardmäßig in der ersten Ebene kleine ausgefüllte Kreise, in der zweiten Ebene Spiegelstriche, in der dritten ein Sternchen und in der vierten Ebene ein Punkt verwendet. Der Text wird wie sonst auch im Blocksatz umbrochen, gleichzeitig werden die Einträge gegenüber dem normalen Text eingerückt.

```
\begin{itemize}
\item Text des ersten Punktes
\item Zweiter Punkt
\end{itemize}
```

Die Marken können auf Wunsch auch geändert werden; soll z. B. in der ersten Ebene ein Spiegelstrich und in der zweiten Ebene ein Plus erscheinen, so kann dies mittels

```
\renewcommand{\labelitemi}{--}
\renewcommand{\labelitemii}{+}
```

geschehen (die dritte und vierte Ebene werden dabei nicht verändert). Mit `\renewcommand` wird ein bestehendes Kommando durch etwas Neues, in diesem Fall durch – und +, ersetzt. Diese Kommandos können auch in der Präambel stehen, wenn sie für das ganze Dokument global gelten sollen. Näheres dazu finden Sie weiter unten im Text. Für alle vier Ebenen gibt es die passende Variable:

`\labelitemi` für die erste Ebene,
`\labelitemii` für die zweite Ebene,
`\labelitemiii` für die dritte Ebene und
`\labelitemiv` für die vierte Ebene.

Wenn statt `itemize` der Umgebungsname `enumerate` verwendet wird, verwendet LaTeX statt der Aufzählungspunkte Zahlen (1., 2. etc.). Beide Umgebungen können ineinander verschachtelt werden. Dabei werden je nach Schachtelungstiefe unterschiedliche Symbole für die Aufzählungspunkte bzw. unterschiedliche Nummerierungsziffern (Kleinbuchstaben, römische Ziffern etc.) verwendet. Außerdem werden die Aufzählungspunkte unterschiedlich stark eingerückt. Auch hier ist eine Verschachtelung bis zur Tiefe 4 möglich. Als Marken werden standardmäßig in der ersten Ebene arabische Ziffern, gefolgt von einem Punkt, in der zweiten Ebene kleine Buchstaben in runden Klammern, in der dritten kleine römische Ziffer, gefolgt von einem Punkt und in der vierten Ebene große Buchstaben, gefolgt von einem Punkt verwendet.

Wie oben beschrieben können auch hier die Labels umdefiniert werden. Es steht

`\labelnumi` für die erste Ebene,
`\labelnumii` für die zweite Ebene,
`\labelnumiii` für die dritte Ebene und
`\labelnumiv` für die vierte Ebene.

Die zugehörigen Zähler heissen `enumi` für den Zähler der ersten Ebene, `enumii` für die zweite Ebene, `enumiii` für die dritte und `enumiv` für die vierte Ebene. Mögliche Darstellungsarten sind:

`\arabic{zähler}` für arabische Ziffern (1, 2, 3, 4, ...),
`\roman{zähler}` für kleine römische Ziffern (i, ii, iii, iv, ...),
`\Roman{zähler}` für große römische Ziffern (I, II, III, IV, ...),
`\alph{zähler}` für kleine Buchstaben (a, b, c, d, ...) und
`\Alph{zähler}` für große Buchstaben (A, B, C, D, ...).

Beispiel:

```
\renewcommand{\labelenumi}{\Alph{enumi}.}
\renewcommand{\labelenumii}{\Roman{enumii}.}
```

Mit der `description`-Umgebung kann eine Aufzählung mit eigenen Marken erreicht werden. Die Aufzählungspunkte werden durch `\item[marke]` festgelegt. Die `marke` kann dabei weggelassen werden. Ansonsten wird sie fett gedruckt. Beispiel:

```
\begin{description}
```

```
\item[Erste Marke] Der Text zur ersten Marke
\item[Zweite Marke] Der Text zur zweiten Marke
\item Text zur dritten Marke fehlt
\end{description}
```

Erste Marke Der Text zur ersten Marke

Zweite Marke Der Text zur zweiten Marke

Der Text zur dritten Marke fehlt

8.2.13 Boxen und Rahmen

Die *Box* spielt bei TeX und LaTeX eine entscheidende Rolle. Buchstaben, Zeilen, Seiten – einfach alle Elemente eines Dokuments – sind aus Boxen zusammengesetzt, die normalerweise unsichtbar sind. Intern baut TeX aus Buchstabenboxen Wortboxen auf, aus diesen Zeilenboxen, aus diesen Absatzboxen, aus diesen mit Seitenkopf und -fuß die Seitenboxen. Es gibt aber auch Boxen, die der Benutzer verwenden kann. Sie enthalten meist Text oder andere Satzelemente und – das Wichtigste – sie werden wie ein einzelnes Zeichen behandelt. LaTeX stellt dem Anwender drei Arten von Boxen zur Verfügung: *LR-Boxen* für Boxen, in denen der Inhalt horizontal angeordnet ist, *PAR-Boxen* für Boxen, die ganze Absätze enthalten und *RULE-Boxen* für Boxen, die nur Farbe enthalten. Boxen können auch ineinander verschachtelt werden, aber eine Box kann nicht am Zeilen- oder Seitenende umbrochen werden.

LR-Boxen

LR-Boxen	
`\mbox{text}`	für eine einfache LR-Box
`\fbox{text}`	für eine einfache gerahmte LR-Box
`\makebox[breite][pos]{text}`	für eine LR-Box
`\framebox[breite][pos]{text}`	für eine gerahmte LR-Box

Der `\mbox`-Befehl dient nicht nur dazu, ein Wort zusammenzuhalten, sondern auch, um beispielsweise im Mathematik-Modus einen „normalen“ Text in eine Formel einzufügen. Mit dem `\fbox`-Befehl kann man einen Textteil einrahmen, worauf wir weiter unten noch eingehen. Als Position kann „l“ für „linksbündig“ und „r“ für „rechtsbündig“ angegeben werden, die Standardeinstellung ist „zentriert“.

Mit `\raisebox{lift}[oberlänge][unterlänge]{text}` kann eine Box analog zu `\mbox` erzeugt werden, die aber gegenüber der Grundlinie um das Maß *lift* angehoben wird. Ein negatives Maß senkt die Box ab, z. B. hoch oder tief. Optional kann man eine Ober- und Unterlänge der entstehenden Box angeben.

PAR-Boxen

PAR-Boxen werden mit dem Kommando `\parbox{breite}[position]{text}` oder analog mit der unten behandelten `minipage`-Umgebung erzeugt. Innerhalb der Box wird der Absatz ganz normal formatiert. Bei der Position kann „t“ oder „b“ angegeben werden, dabei wird bei „t“ die unterste Zeile der Box mit der aktuellen Zeile ausgerichtet, bei „b“ die oberste. Ohne Angabe wird die Box vertikal zentriert zur laufenden Zeile ausgerichtet. Der folgende Absatz zeigt ein Beispiel:

```
\parbox{35mm}{Dies ist eine 30 mm breite Box.
              Sie erscheint vertikal zentriert.}
\hfill die laufende Zeile \hfill
\parbox{55mm}{Dies ist eine 50 mm breite Box.
              Sie erscheint vertikal zentriert.}
```

Dies ist eine 30 mm breite Box. Sie erscheint vertikal zentriert.	die laufende Zeile	Dies ist eine 50 mm breite Box. Sie erscheint vertikal zentriert.

RULE-Boxen

RULE-Boxen werden mit dem Kommando `\rule[lift]{breite}{höhe}` erzeugt. Der Wert von *lift* gibt an, wie weit diese RULE-Box über die Grundlinie angehoben werden soll; negative Maßangaben senken die RULE-Box gegen die Grundlinie ab. Als Standardwert wird 0 mm angenommen. Auch dazu ein Beispiel:

Hier in dieser Zeile taucht ein schwarzes Rechteck der Breite von 10 mm und der Höhe von 3 mm auf ■. Eine RULE-Box der Breite 0 mm ist erlaubt!

Damit lassen sich Stützen konstrieren, wie z. B. Text ohne Stütze und Text mit Stütze (erzeugt durch `\fbox{\rule[-4mm]{0mm}{10mm}Text}`). Die umgebende Box wird im zweiten Fall vertikal ausgedehnt! Auch andere Tricks lassen sich mit RULE-Boxen vollführen, etwa ein kleines graues Kästchen als erstes Aufzählungszeichen:

```
\renewcommand\labelitemi {\textcolor[gray]{0.5}{\rule{1.0ex}{1.0ex}}}
```

Rahmen

Zur Hervorhebung von Texten können diese in einen Rahmen gestellt werden. Das dazu erforderliche Kommando lautet `\fbox{text}` und liefert Resultate wie diesen Text. Wenn mehrzeilige Blöcke eingerahmt werden sollen, muss mit `\fbox` eine `minipage`-Umgebung definiert werden.

Im folgenden Beispiel wird ein kleiner Textblock zentriert und mit `\fbox` eingerahmt. Beim Einsatz von `\hfill` muss beachtet werden, dass damit nur ein Füllabstand zwischen vorhandenen Objekten eingefügt werden kann. Daher muss vor und nach `\hfill` mit `\hbox{}` eine leere Textbox angegeben werden.

```
\hbox{}          % leere Box links
\hfill           % Füllabstand zur minipage
\fbox{
\begin{minipage}{5cm}
Hier wurde eine 5 cm breite Minipage durch ein vor-
und ein nachgestelltes {\tt\textbackslash{}hfill}-Kom\-man\-do
zentriert und mit {\tt\textbackslash{}fbox} eingerahmt.
\end{minipage}}
\hfill           % Füllabstand zur nächsten Box
\hbox{}          % leere Box
```

Hier wurde eine 5 cm breite Minipage durch ein vor- und ein nachgestelltes `\hfill`-Kommando zentriert und mit `\fbox` eingerahmt.

Innerhalb von `\fbox` darf das Kommando `\verb_text_` nicht verwendet werden. Diese Einschränkung können Sie mit der Umgebung `lrbox` umgehen.

Box-Register

Mit `\newsavebox{\name}` wird eine Variable für eine Box mit dem angegebenen Namen angelegt. Diese können Sie mittels `\sbox{\name}{text}` oder `\savebox{\name}[breite][position]{text}` mit einem Inhalt versehen. Der Unterschied zwischen diesen beiden Kommandos ist wie zwischen `\mbox` und `\makebox`. Der Inhalt der Box wird mit dem Befehl `\usebox{\name}` ausgegeben:

```
\newsavebox{\ownbox}
\sbox{\ownbox}{ganz toller Text}
Hier kommt ein \usebox{\ownbox}.
\savebox{\ownbox}[3cm][l]{Text}
Hier kommt ein \usebox{\ownbox}.
```

Hier kommt ein ganz toller Text. Hier kommt ein Text   .

Mehrspaltiger Text

Bereits auf Seite 347 wurde beschrieben, dass durch die Option `twocolumn` im Kommando `\documentclass` der gesamte Text zweispaltig angeordnet werden kann. Für viele Problemstellungen ist diese Lösung aber ungeeignet. Oft sollen nur kleine Textportionen nebeneinander angeordnet werden, während der

restliche Text einspaltig über die volle Breite geht. Auch für solche Situationen bietet LATEX mehrere Varianten an, von denen hier wiederum nur die wichtigste vorgestellt wird: der Umgang mit der `minipage`-Umgebung. Die Syntax sieht folgendermaßen aus:

minipage-Umgebung

```
\begin{minipage}[t]{4cm}  % die erste, 4 cm breite Minipage
text ...                  % der Text in der ersten Minipage
\end{minipage}
\hfill                    % Abstand zwischen den beiden Minipages
\begin{minipage}[t]{4cm}  % die zweite Minipage
text ...
\end{minipage}
\hfill
\begin{minipage}[t]{4cm}  % die dritte Minipage
text ...
\end{minipage}
```

Die Breite aller angegebenen `minipage`-Umgebungen darf die gesamte Textbreite nicht überschreiten. Der optionale Parameter `t` (top) bewirkt, dass die Textblöcke vertikal an der oberen Kante ausgerichtet werden. Alternativ könnten Sie `b` (bottom) angeben, um die Textblöcke an der unteren Kante auszurichten, oder ganz auf den Parameter verzichten, um die Blöcke vertikal zu zentrieren. Das Kommando `\hfill` zwischen den Blöcken bewirkt, dass der verbleibende horizontale Freiraum zwischen ihnen verteilt wird.

```
\begin{minipage}[t]{5.5cm}
Das ist der erste 5,5 cm breite Textblock.
\end{minipage}
\hspace{0.5cm}

\begin{minipage}[t]{5.5cm}
Der zweite Block wird daneben platziert und ist ebenso breit. Die
beiden Blöcke werden an ihrer oberen Kante ausgerichtet.
\end{minipage}
```

Das Ergebnis haben wir zur Verdeutlichung mit einer Box umgeben:

Das ist der erste 5,5 cm breite Textblock.	Der zweite Block wird daneben platziert und ist ebenso breit. Die beiden Blöcke werden an ihrer oberen Kante ausgerichtet.

8.3 Gestaltung wissenschaftlicher Texte

Prinzipiell können Sie jeden beliebigen Text mit LATEX setzen. Seine Vorteile gegenüber anderen Satz- und Textverarbeitungsprogrammen spielt LATEX aber

erst bei der Gestaltung wissenschaftlicher Texte oder bei Büchern aus, bei denen es darum geht, mit geringem Aufwand Inhalts-, Abbildungs-, Literatur- und Stichwortverzeichnisse zu erstellen, Querverweise zu verwenden, Fußnoten einzufügen etc.

8.3.1 Die Titelseite

Die Makros der Titelseite hängen sehr von dem gewählten Style ab und sind eigentlich nur bei „book“ oder „article“ interessant. Für die Gestaltung der Titelseite bieten sich folgende Makros an:

`\title`: Titel des Dokumentes
`\author`: Autor des Dokumentes
`\date`: Datum der Veröffentlichung

Alle diese Befehle sind optional und müssen vor dem eigentlichen Dokument (`\begin{document}`) angegeben werden. Um die Titelseite auszugeben, wird dann der Befehl `\maketitle` verwendet. Beim Befehl `\date` kann entweder `\today` für das aktuelle Datum, eine beliebige Datumsangabe oder gar nichts angegeben werden.

Die Zusammenfassung (Abstract) wird in der Umgebung `abstract` angegeben und steht dann unter dem Titel. Diese Umgebung steht nicht in der Dokumentenklasse „book“ zur Verfügung. Hier ein Beispiel für den Vorspann zu einem Artikel:

```
\documentclass{article}
\usepackage[latin1]{inputenc}

\author{Michael Kofler und Jürgen Plate}
\title{Linux für Studenten}
\date{23.03.2006}

\begin{document}
\maketitle

\begin{abstract}
Diese Abhandlung beschäftigt sich mit der Aufzucht und
Pflege kleiner, elektronischer Pinguine. Als Umgebung
haben sich {\bf Linux} und \LaTeX\ bewährt. In Umgebungen
mit zu vielen Windows neigen die kleinen TUXe zu allergischen
Reaktionen.
\end{abstract}

...

\end{document}
```

Das Ergebnis ist in Abbildung 8.3 zu bewundern:

Abbildung 8.3: Eine LATEX-Titelseite mit Abstract

Titelseite	
`\author`	definiert den Autor
`\title`	definiert den Titel
`\date`	definiert das Veröffentlichungsdatum
`\maketitle`	erzeugt die Titelseite im Dokument
`abstract`	Umgebung für die Zusammenfassung

8.3.2 Bearbeitung umfangreicher Texte

Wenn Sie längere Texte mit LATEX schreiben möchten, ist es sinnvoll, den Text in mehrere Dateien zu zerlegen. Bewährt hat sich dabei die Unterteilung in eine zentrale Steuerungsdatei (z. B. `buch.tex`), eine Datei mit globalen Definitionen und Einstellungen (Stylefile, z. B. `buch.sty`) sowie je eine Datei für jedes Kapitel. (LATEX hat übrigens auch bei solcherart zusammengesetzten Dokumenten keinerlei Probleme mit dem Inhalts- und Stichwortverzeichnis. Querverweise können ohne weiteres von einem Kapitel in ein anderes verweisen.)

Das wichtigste Kommando zum Zusammensetzen von Texten aus mehreren Dateien lautet `\input{datei}`. Es liest an der Stelle seines Auftretens die angegebene Datei und verarbeitet sie, als stünde der darin enthaltene Text an der Stelle des `\input`-Kommandos. `\input` darf auch verschachtelt auftreten. Wenn in `\input` keine Dateikennung angegeben wird, hängt LATEX selbstständig `.tex` an den Dateinamen an.

externe Datei einlesen	
`\input{datei}`	Fügt die angegebene Datei als Teil des Quelltextes an der Stelle des `input`-Befehls ein. Im Gegensatz zum `include`-Befehl erfolgt kein Seitenvorschub.
`\include{datei}`	Erzeugt ein `\newpage` und fügt danach die angegebene Datei an der Stelle des `include`-Befehls ein

Die Zerlegung langer Texte in mehrere Dateien hat nicht nur den Vorteil einer besseren Übersichtlichkeit, sie beschleunigt auch das Arbeitstempo. Während der Arbeit an einem Kapitel können alle anderen `\include`-Zeilen in `buch.tex` durch ein vorangestelltes `%`-Zeichen auskommentiert werden. Die Bearbeitung des Textes durch LaTeX wird dadurch erheblich beschleunigt. Die Dateien `buch.tex` und `buch.sty` sehen in der Regel etwa so aus:

```
% buch.tex: zentrale Steuerungsdatei
\documentclass{book}
\usepackage[fleqn]{amsmath}
\usepackage{amssymb}
\usepackage{alltt}
\usepackage{latexsym}
\usepackage{makeidx}
\usepackage{textcomp}

\usepackage{buch}          % eigenes Zeug

\makeindex
\begin{document}

\include{Text/preface}
\tableofcontents
\newpage
\include{Text/kap1}
\include{Text/kap2}
...
\include{Text/kap19}
\printindex
\end{document}
```

Das Stylefile (Endung `.sty`) enthält ganz normale LaTeX-Anweisungen, aber auch Definitionen von Makros und Umgebungen – eben alles, was man so braucht. Es beginnt mit der Zeile `\ProvidesPackage{Name des Pakets}[Erstellungsdatum]`. Die `RequirePackage`-Zeilen legen fest, welche Pakete ggf. benötigt werden.

```
\ProvidesPackage{buch}[2005/08/31]

\RequirePackage{german}
```

```
\RequirePackage[latin1]{inputenc}
\RequirePackage{verbatim}
\RequirePackage{eurosym}      % Euro-Symbol

% -------------------------------------- Allgemeine Formatanweisungen
\setlength{\parskip}{4pt}% Einzug am Begin des Absatzes
\parindent 0pt        % verzichte auf Einrücken der ersten Zeile

\widowpenalty=10000   %Schusterjungen und Hurenkinder möglichst
\clubpenalty=10000    %ausschließen
\sloppy               %Anzahl der Warnungen reduzieren (es bleiben genug!)

...
```

8.3.3 Inhaltsverzeichnis

Das Inhaltsverzeichnis kann mit dem Kommando `\tableofcontents` an jeder beliebigen Stelle in den Text eingefügt werden. Üblicherweise wird das Inhaltsverzeichnis am Anfang des Texts oder nach dem Vorwort platziert.

Zur Erstellung des Inhaltsverzeichnisses verarbeitet LaTeX die Datei `name.toc`. In diese Datei trägt LaTeX bei jedem Durchlauf mit dem Kommando `\tableofcontents` alle Informationen für das Inhaltsverzeichnis ein (Kapitel-, Abschnitts- und Teilabschnittsnamen zusammen mit ihren Seitennummern). Diese Vorgehensweise hat zur Folge, dass LaTeX im ungünstigsten Fall *dreimal* ausgeführt werden muss, bis das Inhaltsverzeichnis korrekt ist.

Beim ersten Mal existiert `name.toc` noch nicht, d. h. es kann kein Inhaltsverzeichnis erzeugt werden. Beim zweiten Mal kann zwar ein Inhaltsverzeichnis angelegt werden, da dieses aber zumeist ebenfalls einige Seiten beansprucht, verschieben sich alle Seitennummern! Erst beim dritten Durchlauf stimmen die Seitennummern im Inhaltsverzeichnis mit den tatsächlichen Seitennummern überein.

In das Inhaltsverzeichnis werden normalerweise die Texte aller vorhandenen `\part`-, `\chapter`-, `\section`-, `\subsection`- und `\subsubsection`-Kommandos aufgenommen. Die Anzahl der Gliederungsebenen kann mit `\setcounter{tocdepth}{n}` vermindert werden (siehe Seite 395).

Die Formatierung des Inhaltsverzeichnisses erfolgt automatisch. LaTeX wählt dabei passende Schriftgrößen, rückt untergeordnete Einträge ein und füllt den Zeilenfreiraum zwischen dem Eintrag und der Seitennummer mit Punkten. Das Inhaltsverzeichnis dieses Buchs ist kein typisches Beispiel, weil hier das Standardlayout von LaTeX aus ästhetischen Gründen verändert wurde. Eine bessere Vorstellung, wie ein Inhaltsverzeichnis von LaTeX normalerweise aussieht, bietet die Abbildung **??** auf Seite **??**.

Inhaltsverzeichnis

`\tableofcontents`	fügt an dieser Stelle das Inhaltsverzeichnis ein

8.3.4 Querverweise

Querverweise im Buch werden mit den drei Kommandos `\label`, `\ref` und `\pageref` erstellt. Welche Nummer `\ref` liefert, hängt von dem Ort ab, an dem `\label` ausgeführt wurde: In der Regel handelt es sich um eine Abschnittsnummer, die den aktuellen Abschnitt bezeichnet. `\label` kann aber auch in Umgebungen wie `equation`, `figure` oder `table` verwendet werden. `\ref` liefert in diesem Fall die Nummer der Formel, Abbildung, Tabelle etc.

Mit Querverweisen verhält es sich ähnlich wie mit den Seitenzahlen des Inhaltsverzeichnisses: Sie werden der Datei `name.aux` entnommen, die beim letzten LaTeX-Durchlauf erstellt wurde. Nach Änderungen in einem LaTeX-Text sind daher wie oben mindestens zwei Durchläufe notwendig, bis alle Seitennummern korrekt sind.

Abschließend ein Beispiel: Wenn an dieser Stelle im Text ein Markierungspunkt mit `\label{verweis-bsp}` erstellt wird, dann liefert `\ref{verweis-bsp}` das Ergebnis 8.3.4 und `\pageref{verweis-bsp}` die Seitennummer 375.

Querverweise	
`\label{marke}`	definiert einen Markierungspunkt
`\pageref{marke}`	liefert die Seitennummer der Markierung
`\ref{marke}`	liefert Abschnitts-, Abbildungs- oder Tabellennummer

8.3.5 Fußnoten

Fußnoten werden mit `\footnote{text}` erstellt. LaTeX fügt daraufhin an dieser Stelle eine hochgestellte Nummer als Fußnote ein und platziert den Fußnotentext an das Ende der laufenden Seite.

Fußnoten werden automatisch durchnummeriert, wobei die Nummerierung bei den Texttypen *book* und *report* in jedem Kapitel neu beginnt. Das Aussehen von Fußnoten ist in Abbildung **??** auf Seite **??** dokumentiert.

Fußnoten	
`\footnote{text}`	fügt eine neue Fußnote in den Text ein

8.3.6 Der Anhang

Der Anhang wird mit `\begin{appendix}` eingeleitet und mit `\end{appendix}` beendet. Sie haben hier die gleichen Möglichkeiten der Gestaltung wie im Dokumententext, im Anhang werden die Kapitel jedoch alphabetisch nummeriert.

Anhang

```
\begin{appendix}

 ...        % alle Kapitel des Anhangs

 \end{appendix}
```

8.3.7 Literaturverzeichnis

Die Verwaltung des Literaturverzeichnisses erfolgt in zwei Schritten: Zuerst muss am Ende des Buchs (dort, wo das Literaturverzeichnis erscheinen soll) eine Liste mit allen Einträgen erstellt werden. Anschließend kann im gesamten Text auf die Einträge dieses Verzeichnisses verwiesen werden.

Literaturverzeichnis

```
% im laufenden Text
\cite{marke1}          % Verweis auf den Eintrag 'marke1'

% am Ende des Artikels/Buchs
\begin{thebibliography}{n}
\bibitem{marke1} Text1  % Autor, Titel etc.
\bibitem{marke2} Text2
\end{thebibliography}
```

LaTeX erzeugt an der Stelle der `thebibliography`-Umgebung eine Liste, in der alle Einträge in eckigen Klammern durchnummeriert werden ([1], [2] etc.). Der Parameter n gilt als Maß für die Einrückung der Einträge. Für bis zu neun Einträge kann für n die Zahl 9 eingesetzt werden, für bis zu 99 Einträge die Zahl 99 etc. Die `\bibitem`-Einträge werden nicht automatisch sortiert – Sie müssen sich also selbst um eine geeignete Ordnung kümmern. Die Einträge werden auch nicht automatisch formatiert. Sie können aber alle Kommandos zur Einstellung der Schriftart verwenden und so den Namen des Autors fett, den Titel kursiv etc. formatieren.

Im laufenden Text können Sie nun mit `\cite{marke}` auf einen Eintrag im Literaturverzeichnis verweisen. LaTeX setzt an dieser Stelle im Text eine eckige Klammer mit der entsprechenden Nummer ein.

Während das `\bibitem`-Kommando für gelegentliche Publikationen vollkommen ausreichend ist, bietet LaTeX eine noch viel leistungsstärkere Alternative

durch das Zusatzprogramm *bibtex*. Die vollständige Liste aller jemals (etwa in einer ganzen Abteilung) verwendeten Referenzen wird in einer eigenen Datei gespeichert. In der eigentlichen Publikation kann darauf durch Kürzel verwiesen werden. *bibtex* wertet diese Informationen aus und erstellt automatisch ein Quellenverzeichnis, das nur die tatsächlich genutzten Referenzen enthält. Dieses Zusatzprogramm ist im LaTeX-Begleiter von Michel Goosens et al. ausführlich beschrieben.

Das Quellenverzeichnis dieses Buchs wurde übrigens nicht mit den gerade beschriebenen Methoden erstellt. Die wenigen Quellen erfordern keine Nummerierung und die meisten Titel beziehen sich auf weiterführende oder vertiefende Literatur.

8.3.8 Stichwortverzeichnis

Wenn Sie Ihren Text mit einem Stichwortverzeichnis ausstatten möchten, dann sind dazu mehrere Arbeitsschritte erforderlich: Sie müssen mit `\usepackage` das Package *makeidx* laden und vor `\begin{document}` das Kommando `\makeindex` ausführen. Die Indexeinträge müssen im Text mit dem Kommando `\index` markiert werden. Schließlich muss an der Stelle im Text, an der das Stichwortverzeichnis erscheinen soll, das Kommando `\printindex` angegeben werden.

Doch damit nicht genug: Nachdem die so präparierte Textdatei zum ersten Mal mit LaTeX bearbeitet wurde, muss das Linux-Kommando `makeindex name.idx` ausgeführt werden. Dabei wird die von LaTeX erstellte Datei `name.idx` mit allen Indexeinträgen ausgewertet und sortiert (mit der Option `-g` sogar nach der deutschen Sortierordnung). Das Ergebnis wird in der Datei `name.ind` gespeichert. Beim nächsten LaTeX-Durchlauf wird diese Datei bei der Ausführung von `\printindex` eingelesen.

Die wichtigsten Syntaxvarianten von `\index{eintrag}` gehen aus der folgenden Tabelle hervor. Mit den Zeichenkombinationen `|(` und `|)` lassen sich Seitenbereiche angeben. Der resultierende Indexeintrag sieht dann beispielsweise so aus: Eintrag 34-38. Ein nach `|` und *ohne* vorangestelltes \-Zeichen angegebenes Kommando kann zur Formatierung der Seitenzahl eingesetzt werden. Ein Muster für die Definition eines geeigneten Kommandos wurde mit `\ii` angegeben. `@` kann dazu eingesetzt werden, auch Formeln oder speziell sortierte Einträge richtig im Indexverzeichnis einzuordnen. Alle genannten Formatierungsmethoden können auch kombiniert werden, beispielsweise um einen Subeintrag besonders hervorzuheben.

Einträge in das Stichwortverzeichnis

```
\usepackage{makeidx}                % nach \documentclass
...
\makeindex                          % vor \begin{document}
\newcommand{\ii}[1]{{\it #1}}       % kursive Seitenzahlen
...
\index{Eintrag}                     % normaler Indexeintrag
\index{Haupteintrag!Subeintrag}     % Subeintrag
\index{Haupt!Sub!Subsub}            % Subsubeintrag
\index{Eintrag|(}                   % Eintrag von Seite
\index{Eintrag|)}                   % Eintrag bis Seite
\index{Eintrag|ii}                  % Seitennummer kursiv
\index{Pi@$\pi$}                    % Formel wie 'Pi' sortieren
\index{Eintrag@{\bf Eintrag}}       % fett, richtig  sortieren
...
\printindex                         % vor \end{document}
```

Abschließend folgen noch einige Beispiele, die die Wirkungsweise des `\index`-Kommandos demonstrieren. Die Ergebnisse können Sie sich im Stichwortverzeichnis am Ende des Buchs ansehen.

```
\index{Indexbeispiel}
\index{Indexbeispiel!Subeintrag}
\index{Indexbeispiel!Subeintrag@{\tt Subeintrag Courier}}
\index{Indexbeispiel!Sonderz@{\verb?Sonderzeichen % \?}}
\index{Indexbeispiel!kursiver Eintrag@{\it kursiver Eintrag}}
\index{Indexbeispiel!kursive Seitenziffer|ii}
\index{index@{\verb?\index?}}
\index{printindex@{\verb?\printindex?}}
\index{makeindex@{\verb?makeindex?}}
\index{usepackage@\verb?\usepackage?!makeidx@{\tt makeidx}}
\index{Stichwortverzeichnis!LaTeX}
\index{Index}
```

Zum Programm `makeindex` existiert eine detaillierte Beschreibung als `man`-Seite. Der Text hat allerdings den Nachteil, dass er nicht LaTeX-spezifisch ist. (`makeindex` kann auch zur Bearbeitung von Indexdateien anderer Programme eingesetzt werden.) Eine typische Steuerdatei für das Programm sieht folgendermaßen aus:

```
preamble
 "\\begin{theindex}\n
  \\thispagestyle{empty}\n"
postamble
 "\\end{theindex}\n"
delim_0 "\\hspace{1mm} \\dotfill "
delim_1 ", "
delim_2 ", "
delim_r "-"
delim_t
```

```
item_0 "\n \\item "
item_1 "\n \\subitem "
item_2 "\n \\subsubitem "
item_01 "\n \\subitem "
item_x1 "\n \\subitem "
item_12 "\n \\subsubitem "
item_x2 "\n \\subsubitem "
indent_space
indent_length 0
group_skip "\n\n \\indexspace\n"
encap_prefix "\\"
encap_infix "{"
encap_suffix "}"
headings_flag 1
symhead_positive "Symbole"
symhead_negative "symbole"
numhead_positive "Ziffern"
numhead_negative "ziffern"
heading_prefix "\n \\vspace{2pt} {\\Large "
heading_suffix "} \\nopagebreak \\vspace{1pt} \n"
```

Wenn die Steuerdatei `myind.sty` heißt und die LaTeX-Datei `linuxbuch.tex`, wird der Index mit `makeindex -s myind.sty linuxbuch` generiert.

8.4 Abbildungen

Um eine Abbildung in ein LaTeX-Dokument einzubinden, werden zumeist eine `figure`-Umgebung und ein `\includegraphics`-Kommando kombiniert. Die `figure`-Umgebung ist für die Platzierung und Beschriftung der Abbildung verantwortlich, während das Kommando `\includegraphics` aus dem `graphicx`-Paket eine PostScript-Grafikdatei einliest.

Die `figure`-Umgebung ist auch wieder ein Gleitobjekt (wie bei den Tabellen, Seite 363 beschrieben). Der optionale Parameter dieser Umgebung bestimmt, wie die Grafik im Text platziert wird. Dabei gilt auch für die `figure`-Umgebung das dort gesagte: Als Parameter ist entweder `h` (*here*), `t` (*top*), `b` (*bottom*) oder `p` (*page*) erlaubt – oder eine beliebige Kombination der Buchstaben. Wenn LaTeX dennoch die Abbildung woanders platziert, gibt ein zusätzliches Ausrufezeichen Ihrem Platzierungswunsch mehr Nachdruck (also z. B. `[h!]`).

Platzierung der Abbildung

- `h:` Die Abbildung wird genau an dieser Stelle im Text angezeigt.
- `t:` Die Abbildung wird am Beginn der laufenden Seite platziert.
- `b:` Die Abbildung wird am Ende der laufenden Seite platziert.
- `p:` Mehrere Abbildungen werden auf einer eigenen Seite zusammengefasst.

In zweispaltigen Texten kann statt `figure` der Umgebungsname `figure*` verwendet werden. In diesem Fall erstreckt sich die Abbildung über beide Spalten (anstatt per Default nur die Breite einer Spalte zu nutzen). `figure*` kann nicht mit der Option `h` kombiniert werden.

Zur Beschriftung der Grafik wird `\caption` eingesetzt. LaTeX stellt dem eigentlichen Beschriftungstext „Abbildung n:" voran, wobei für n entweder eine laufende Nummer (Texttyp *article*) oder eine Kapitelnummer (3.5 für das fünfte Bild im dritten Kapitel) eingesetzt wird. Wenn innerhalb von `\caption` eine `\label`-Anweisung verwendet wird, kann mit `\ref` auf die Abbildungsnummer und mit `\pageref` auf die dazugehörige Seitennummer verwiesen werden.

figure-Umgebung

```
\begin{figure}[h]

% Kommandos zur Erzeugung der eigentlichen Grafik z. B.
% \includegraphics{...} oder den internen Grafikmöglichkeiten

\caption{\label{marke-fuer-querverweis}Beschriftungstext}
\end{figure}
```

Der Beschriftungstext wird automatisch zentriert. Wenn das Bild schmaler als der Beschriftungstext ist, sollte dieser mit `\parbox{7cm}{\caption{...}}` auf die Breite des Bildes (hier 7 cm) beschränkt werden.

Das graphicx-Paket: Das *graphicx*-Paket stellt unter anderem das Kommando `\includegraphics` zur Verfügung. Damit kann eine Grafikdatei in das LaTeX-Dokument eingebunden werden. Die Syntax für `\includegraphics` sieht so aus:

```
\usepackage{graphicx}
...
\includegraphics[option1=... option2=...]{filename}
```

Die Optionen steuern unter anderem die Größe, Skalierung und Rotierung der Grafik. Die folgende Tabelle fasst die wichtigsten Optionen zusammen.

includegraphics-Optionen

`width=`n	bestimmt die Breite der Grafik im Dokument
`height=`n	bestimmt die Höhe der Grafik
`scale=`n	gibt an, wie stark die Grafik skaliert werden soll (relativ zur Originalgröße der Grafik)
`angle=`n	gibt an, um wie viel Grad die Grafik gedreht werden soll

Zur Größenangabe ist normalerweise die Angabe einer Option ausreichend (`width` oder `height`) – die Grafik wird dann automatisch korrekt skaliert. Bei `width` kann man auch wieder auf die aktuelle Textbreite zurückgreifen, z. B. `width={\textwidth}` oder `width={0.95 \textwidth}`.

Wenn Sie das LaTeX-Dokument später in das PostScript-Format umwandeln (`dvips`) und ausdrucken möchten, kommen für `\includegraphics` nur PostScript-Dateien (EPS, PS) in Frage. Grafiken in anderen Formaten (GIF, TIFF, JPEG, PNG) müssen Sie gegebenenfalls vorher mit einem Bildverarbeitungsprogramm in das EPS-Format umwandeln. Der direkte Import von Bitmap-Grafiken ist nur dann zulässig, wenn Sie Ihr LaTeX-Dokument mit `pdflatex` direkt in eine PDF-Datei umwandeln möchten (siehe Seite 425).

Abbildungen werden generell weder zentriert noch eingerahmt. Wenn diese Formatierungsmerkmale erwünscht sind, können Sie das Bild mit `\begin{center}` ...`\end{center}` oder mit `\centerline` zentrieren und mit `\fbox` in einen Rahmen setzen.

```
\begin{figure}
\begin{center}
\includegraphics[width=5cm]{figure.ps}
\caption{zentriert}
\end{center}
\end{figure}

\begin{figure}
\begin{center}
\fbox{\includegraphics[width=5cm]{figure.ps}}
\caption{zusätzlich gerahmt}
\end{center}
\end{figure}
```

Wenn mehrere Abbildungen nebeneinander platziert werden sollen, müssen wie im folgenden Beispiel innerhalb der `figure`-Umgebung mehrere Minipages platziert werden. Der umgekehrte Fall – also die Verwendung einer `figure`-Umgebung innerhalb einer Minipage – ist nicht möglich. Aus diesem Grund ist es nicht ohne weiteres möglich, eine beschriftete Grafik und Text nebeneinander zu platzieren. Sie können aber eine unbeschriftete PostScript-Grafik durch die direkte Verwendung des `\includegraphics`-Kommandos ohne `figure`-Umgebung in einer Minipage ausgeben.

```
\begin{figure}[h]
\begin{minipage}[t]{5.5cm}
\begin{center}
\includegraphics[width=5cm]{bilder/b-latex-screenshot.eps}
\caption{\label{latex-fig1} \textsl{Dieser Screenshot ...
  eingebunden.}
\end{center}
\end{minipage}
\hfill
\begin{minipage}[t]{5.5cm}
\begin{center}
\setlength{\fboxsep}{0mm} %kein Abstand zur Umrahmung
\fbox{\includegraphics[width=5cm]{bilder/b-latex-maple.eps}}
```

```
\caption{\label{latex-fig2} \textsl{Dieses Diagramm wurde ...
    eingebunden.}}
\end{center}
\end{minipage}
\end{figure}
```

Abbildung 8.4: *Dieser Screenshot ... eingebunden.*

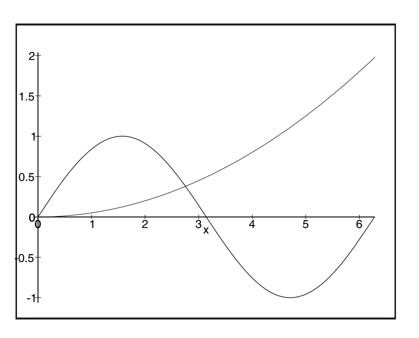


Abbildung 8.5: *Dieses Diagramm wurde ... eingebunden.*

Die Beispielabbildungen 8.4 und 8.5 wurden mit den obigen Kommandos erzeugt. Die Verweise auf die Bildnummern wurden mit `\ref{latex-fig1}` bzw. mit `\ref{latex-fig2}` erzeugt.

Hinweis

Wenn eine PostScript-Grafik nicht an der durch die `figure`-Umgebung vorgesehenen Position erscheint, sondern irgendwo anders auf der Seite, ist meist eine falsche BoundingBox-Information innerhalb der importierten PostScript-Grafik schuld. Die BoundingBox-Zeile gibt die Position und die Größe der Grafik an.

Frühere Versionen von `ksnapshot` produzierten beispielsweise falsche BoundingBoxes, wenn der Screenshot als EPS-Datei gespeichert wird. Speichern Sie den Screenshot stattdessen als Bitmap und verwenden Sie ein anderes Programm, um daraus eine korrekte EPS-Datei zu erzeugen.

Eine Menge weiterer Informationen zum Thema Grafik und Farben finden Sie auf der folgenden, exzellenten Seite im Internet:
http://www.linmpi.mpg.de/ daly/latex/grf.htm

8.5 Mathematische Formeln

Damit in LATEX mathematische Formeln dargestellt werden, müssen Sie in den Mathematik-Modus umschalten. Sämtliche hier vorgestellten Kommandos können nur in diesem Modus ausgeführt werden!

Zur Aktivierung des Mathe-Modus haben Sie drei Möglichkeiten:

- Die Formel wird im laufenden Text zwischen zwei `$`-Zeichen gesetzt. In diesem Fall versucht LaTeX, die Formel an die Zeilenhöhe anzupassen.
- Die Formel wird in einem eigenen Absatz mit `\[` eingeleitet und mit `\]` beendet. Sie steht dann als eigener Absatz zentriert im Text.
- Die Formel wird in einer `equation`-Umgebung geschrieben. Auch hier wird die Formel in einem eigenen Absatz zentriert dargestellt, wobei sie in der `equation`-Umgebung zusätzlich automatisch nummeriert wird.

LaTeX-Formelmodi

`$formel$`	Formel im Fließtext
`\[ formel \]`	eigenständige Formel

`\begin{equation}`
`  formel`
`\end{equation}`
	eigenständige Formel mit Nummerierung

Beachten Sie bitte, dass Formeln im Fließtext anders aussehen als eigenständige Formeln: Bei `$`-Formeln verwendet LaTeX eine etwas kleinere Schriftart und versucht, die Formel so zu setzen, dass sie vertikal möglichst wenig Platz beansprucht. Insbesondere werden Hoch- und Tiefstellungen bei manchen mathematischen Symbolen (Limes, Integrale, Summen) anders platziert.

Im Folgenden sehen Sie dreimal dieselbe Formel `\int_{i=1}^{n} x^i dx`: Im Fließtext sieht die mit `$` geklammerte Formel so aus: $\int_{i=1}^{n} x^i dx$. Mit `\[ ... \]` bzw. zwischen `\begin{equation}` und `\end{equation}` erhalten Sie dagegen folgende Ergebnisse:

$$\int_{i=1}^{n} x^i dx$$

$$\int_{i=1}^{n} x^i dx \tag{8.1}$$

Wenn Sie möchten, dass eigenständige Formeln nicht zentriert, sondern linksbündig dargestellt werden, können Sie am Beginn des LaTeX-Dokuments die Anweisung `\usepackage{fleqn}` verwenden. Anschließend können Sie mit `\mathindent2cm` eine Einrücktiefe vom linken Rand (hier: 2 cm) angeben. Alternativ dazu können Sie mit `\usepackage{leqno}` erreichen, dass LaTeX Formeln in der `equation`-Umgebung nicht rechts-, sondern linksbündig nummeriert.

Innerhalb von Formeln wird Text generell kursiv dargestellt. Für Variablen ist das die übliche mathematische Schreibweise. Funktionsnamen sollen dagegen

zumeist in aufrechter Schrift dargestellt werden. Dazu existieren eigene LaTeX-Kommandos wie `\sin`. $\sin(x^2)$ wird durch den Text `$\sin(x^2)$` erzeugt. LaTeX kennt die folgenden Kommandos zur Ausgabe von Funktionen:

Schlüsselwörter für mathematische Funktionen

```
\arccos, \arcsin, \arctan, \arg, \cos, \cosh, \cot, \coth, \csc,
\deg, \det, \dim, \exp, \gcd, \hom, \inf, \ker, \lg, \lim, \liminf,
\limsup, \ln, \log, \max, \min, \Pr, \sec, \sin, \sinh, \sup, \tan,
\tanh
```

Der Mathematik-Modus beseitigt auch alle Leerzeichen in der Formel und bestimmt die Abstände zwischen den Elementen nach anderen Regeln. Normaler Text wird daher entstellt. Soll er (oder ein oben nicht angeführter Funktionsname) innnerhalb von Formeln dargestellt werden, muss der Text in einer (unsichtbaren) Box stehen, was Sie mit der Anweisung `\mbox{...}` erreichen. Es sind auch Anweisungen wie `{\it varname}` oder `{\bf X}` erlaubt, um längere Variablennamen auszugeben oder Texte hervorzuheben.

LaTeX geht im Mathe-Modus manchmal recht sparsam mit Abständen um. In der Formel $\sin(xy)$ (`\sin(x y)`) ist kaum zu erkennen, dass x und y zwei eigenständige Variablen sind, die miteinander multipliziert werden. Zur Vergrößerung der Abstände können Sie die drei Kommandos `\,` (kleiner Abstand), `\:` (mittlerer Abstand) und `\;` (großer Abstand) einsetzen: `\sin(x \: y)` wird dann zu $\sin(x\:y)$.

Konstruktion mathematischer Formeln

Kommando	Ergebnis	Kommando	Ergebnis
`a^{b}`	a^b	`\sum_{a}^{b} c`	$\sum_{a}^{b} c$
`a_{b}`	a_b	`\prod_{a}^{b} c`	$\prod_{a}^{b} c$
`\frac{a}{b}`	$\frac{a}{b}$	`{a \choose b}`	${a \choose b}$
`\sqrt{a}`	$\sqrt{a}$	`\overline{abc}`	$\overline{abc}$
`\sqrt[n]{a}`	$\sqrt[n]{a}$	`\underline{abc}`	$\underline{abc}$
`\int_{a}^{b} c`	$\int_{a}^{b} c$	`\overbrace{abc}^{d}`	$\overbrace{abc}^{d}$
`\oint_{a}^{b} c`	$\oint_{a}^{b} c$	`\underbrace{abc}_{d}`	$\underbrace{abc}_{d}$

Die oben genannten Kommandos werden zur Zusammensetzung von Brüchen, Wurzeln, Integralen, Summen etc. eingesetzt. Die Kommandos können beliebig verschachtelt werden, um Wurzeln in Brüchen oder Ähnliches zu erreichen. LaTeX kümmert sich um eine geeignete Schriftgröße und andere Formatierungsdetails. Beachten Sie, dass Sie beim Hoch- und Tiefstellen den jeweiligen Ausdruck in geschweifte Klammern setzen – andernfalls gilt ^ bzw. _ nur für das erste nachfolgende Zeichen. Es lassen sich auch recht einfach Formeln aus der Digitaltechnik setzen, beispielsweise ergibt

```
\[ y = (\overline{x1} * \overline{x2} * x3) +
       (\overline{x1} * x2 * \overline{x3)} +
       (x1 * \overline{x2} * \overline{x3}) +
       (x1 * \overline{x2} * x3) + (x1 * x2 * \overline{x3}) \]
```

die Logikgleichung:

$$y = (\overline{x1} * \overline{x2} * x3) + (\overline{x1} * x2 * \overline{x3)} + (x1 * \overline{x2} * \overline{x3}) + (x1 * \overline{x2} * x3) + (x1 * x2 * \overline{x3}).$$

Wenn Sie mit den von LaTeX automatisch gewählten Schriftgrößen nicht zufrieden sind, können Sie diese durch vier Kommandos beeinflussen:

Schriftgröße in mathematischen Formeln	
`\displaystyle`	normal
`\textstyle`	etwas kleiner (Schriftart in $-Formeln)
`\itriptstyle`	noch kleiner (Indizes und Hochzahlen erster Ordnung)
`\itriptscriptstyle`	winzig (Indizes und Hochzahlen zweiter Ordnung)

Die folgenden Formeln demonstrieren das Zusammenspiel von einigen der hier aufgezählten Kommandos. In den Formeln kommen auch einige Kommandos vor, die erst in späteren Abschnitten behandelt werden.

```
\[ \frac{ \frac{a+1}{b-1} }{ \frac{c+1}{d-1}}
   \qquad \mbox{und} \qquad
   \frac{\displaystyle \frac{a+1}{b-1}}
        {\displaystyle \frac{c+1}{d-1}}  \]
```

$$\frac{ \frac{a+1}{b-1} }{ \frac{c+1}{d-1}} \qquad \mbox{und} \qquad \frac{\displaystyle \frac{a+1}{b-1}}{\displaystyle \frac{c+1}{d-1}}$$

```
\[ \oint_C f(z) \, dz = \int_0^{2 \pi} f(z(t)) \,
   \frac{d \, z(t)}{dt} \, dt
   \qquad\mbox{mit}\qquad
   z(t)= z0 + r\, (\cos(t) + I\, sin(t)) \]
```

$$\oint_C f(z)\,dz = \int_0^{2\pi} f(z(t))\,\frac{d\,z(t)}{dt}\,dt \qquad \text{mit} \qquad z(t) = z0 + r\,(\cos(t) + I\,sin(t))$$

```
\[   \left[      {\begin{array}{c}
  {\displaystyle \frac {{\frac {{ \partial}}{{ \partial}{u}}}\,
    {\rm f}(\,{u}, {v}, {z}\,)}{{a}\,\sqrt {{\rm sin}(\,{v}\,)^{2} +
    {\rm sinh}(\,{u}\,)^{2}}}} \\[5mm]
  {\displaystyle \frac {{\frac {{ \partial}}{{ \partial}{v}}}\,
    {\rm f}(\,{u}, {v}, {z}\,)}{{a}\,\sqrt {{\rm sin}(\,{v}\,)^{2} +
    {\rm sinh}(\,{u}\,)^{2}}}} \\[5mm]
  {\frac {{ \partial}}{{ \partial}{z}}}\,{\rm f}(\,{u}, {v}, {z}\,)
\end{array}}    \right]   \]
```

$$\left[{\begin{array}{c}
{\displaystyle \frac {{\frac {{ \partial}}{{ \partial}{u}}}\,{\rm f}(\,{u}, {v}, {z}\,)}{{a}\,\sqrt {{\rm sin}(\,{v}\,)^{2} + {\rm sinh}(\,{u}\,)^{2}}}} \\[5mm]
{\displaystyle \frac {{\frac {{ \partial}}{{ \partial}{v}}}\,{\rm f}(\,{u}, {v}, {z}\,)}{{a}\,\sqrt {{\rm sin}(\,{v}\,)^{2} + {\rm sinh}(\,{u}\,)^{2}}}} \\[5mm]
{\frac {{ \partial}}{{ \partial}{z}}}\,{\rm f}(\,{u}, {v}, {z}\,)
\end{array}} \right]$$

8.5.1 Klammern

Klammern werden prinzipiell direkt mit `(..)` oder `[...]` gebildet. Da geschweifte Klammern in LaTeX eine besondere Bedeutung haben, muss ihnen ein Backslash vorangestellt werden, also `\{...\}`.

In mathematischen Formeln soll die Größe der Klammern im Normalfall dem geklammerten Ausdruck entsprechen. Standardmäßig ist das nicht der Fall. Das Verhalten kann aber mit den Kommandos `\left` und `\right` realisiert werden, die der linken bzw. der rechten Klammer vorangestellt werden. Das vorangegangene Beispiel zeigt die Anwendung dieser beiden Kommandos. `\left` und `\right` können auch vor einigen weiteren mathematischen Symbolen verwendet werden, etwa vor → für Beträge.

Will man nur einseitig eine Klammer setzen (nur rechts oder nur links), gibt es die „unsichtbaren“ Klammern `\left.` und `\right.`, die dafür sorgen, dass wieder Klammerpaare entstehen. In der folgenden Formel wurde die linke Klammer unsichtbar gemacht:

$$
\left.\begin{array}{c}
\frac{\frac{\partial}{\partial u}\,\mathrm{f}(\,u, v, z\,)}{a\,\sqrt{\sin(\,v\,)^2 + \sinh(\,u\,)^2}} \\
\frac{\frac{\partial}{\partial v}\,\mathrm{f}(\,u, v, z\,)}{a\,\sqrt{\sin(\,v\,)^2 + \sinh(\,u\,)^2}} \\
\frac{\partial}{\partial z}\,\mathrm{f}(\,u, v, z\,)
\end{array}\right]
$$

8.5.2 Matrizen

Zur Darstellung von Matrizen wird die `array`-Umgebung verwendet. Sie ähnelt sehr stark der `tabular`-Umgebung, die für Tabellen verwendet wird. Die generelle Syntax lautet:

array-Umgebung

```
\begin{array}{ccc}       % für jede Spalte ein c (centered)
term1 & term2 & term3\\  % Terme durch & trennen, Zeilenende mit \\
term4 & term5 & term6    % in der letzten Zeile kein \\
\end{array}
```

Die `array`-Umgebung produziert ungeklammerte Matrizen. Wenn die Matrix geklammert werden soll, müssen den runden oder eckigen Klammern die Kommandos `\left` bzw. `\right` vorangestellt werden.

```
\[                      % Anfang der Formel
\left (                 % große Klammer auf
\begin{array}{cc}       % zweispaltige Matrix
x^y   & \frac{\alpha}{\beta}\\[2.5mm]
a+b+c & \frac{a+b+c}{x^2}
\end{array}             % Ende der Matrix
\right )                % große Klammer zu
\]                      % Ende der Formel
```

$$
\left(
\begin{array}{cc}
x^y & \frac{\alpha}{\beta}\\[2.5mm]
a+b+c & \frac{a+b+c}{x^2}
\end{array}
\right)
$$

8.5.3 Mathematische Sonderzeichen

Die Zeichen `+ - / = ! ' | ( ) [` und `]` können ohne besondere Umstände direkt in Formeln verwendet werden. Daneben existieren unzählige weitere

Sonderzeichen (Operatoren, Pfeile, mathematische Symbole), die durch LaTeX-Kommandos gebildet werden können. Die folgenden Tabellen geben (auszugsweise) die Kommandos für die wichtigsten Sonderzeichen an. Wenn Sie diese Sonderzeichen im normalen Text (und nicht in einer Formel) verwenden möchten, müssen Sie das Kommando zwischen zwei `$`-Zeichen setzen!

Sonderzeichen

`\infty`	∞	`\cdot`	$\cdot$	`\pm`	$\pm$	`\neq`	$\neq$
`\partial`	∂	`\circ`	$\circ$	`\times`	$\times$	`\sim`	$\sim$
`\Re`	$\Re$	`\bullet`	$\bullet$	`\div`	$\div$	`\simeq`	$\simeq$
`\Im`	$\Im$	`\ldots`	$\ldots$	`\ast`	$\ast$	`\approx`	$\approx$
`\forall`	$\forall$	`\vdots`	$\vdots$	`|`	$\|$	`\equiv`	$\equiv$
`\exists`	$\exists$	`\cdots`	$\cdots$	`\vee`	$\vee$	`\le`	$\le$
		`\ddots`	$\ddots$	`\wedge`	$\wedge$	`\ge`	$\ge$
				`\nabla`	∇	`\ll`	$\ll$
				`\oplus`	$\oplus$	`\gg`	$\gg$
				`\ominus`	$\ominus$		
				`\otimes`	$\otimes$		

Pfeile

`\leftarrow`	$\leftarrow$	`\Leftarrow`	$\Leftarrow$	`\nearrow`	$\nearrow$
`\rightarrow`	$\rightarrow$	`\Rightarrow`	$\Rightarrow$	`\searrow`	$\searrow$
`\uparrow`	$\uparrow$	`\Uparrow`	$\Uparrow$	`\swarrow`	$\swarrow$
`\downarrow`	$\downarrow$	`\Downarrow`	$\Downarrow$	`\nwarrow`	$\nwarrow$
`\leftrightarrow`	$\leftrightarrow$	`\Leftrightarrow`	$\Leftrightarrow$		
`\updownarrow`	$\updownarrow$	`\Updownarrow`	$\Updownarrow$		
`\hookleftarrow`	$\hookleftarrow$				

In Formeln müssen einzelne Variablen häufig durch Vektorpfeile, Ableitungsstriche oder -punkte oder durch andere Zusatzsymbole gekennzeichnet werden. Die folgende Tabelle beschreibt die wichtigsten Markierungsbefehle. Beachten Sie bitte, dass Kommandos wie `\vec` nur für einzelne Buchstaben verwendet werden können (und nicht für Buchstabengruppen oder noch längere Ausdrücke).

Vektoren und Ableitungen

`\bar{x}`	$\bar{x}$	`\tilde{x}`	$\tilde{x}$	`x'`	x'	`x'''`	x'''
`\dot{x}`	$\dot{x}$	`\ddot{x}`	$\ddot{x}$	`\vec{x}`	$\vec{x}$		

8.5.4 Griechische und kalligrafische Buchstaben

Griechische Buchstaben werden ebenfalls durch LaTeX-Kommandos dargestellt. Der Kommandoname ergibt sich aus dem Namen des Buchstabens, also `\alpha` für ein kleines α und `\Delta` für ein großes Δ.

Griechische Buchstaben							
\alpha	α	\kappa	κ	\Psi	Ψ	\Upsilon	Υ
\beta	β	\lambda	λ	\omega	ω	\varepsilon	ε
\chi	χ	\Lambda	Λ	\Omega	Ω	\varphi	φ
\delta	δ	\mu	μ	\rho	ρ	\varpi	ϖ
\Delta	Δ	\nu	ν	\sigma	σ	\varrho	ϱ
\epsilon	ϵ	\phi	ϕ	\Sigma	Σ	\varsigma	ς
\eta	η	\Phi	Φ	\tau	τ	\vartheta	ϑ
\gamma	γ	\pi	π	\theta	θ	\xi	ξ
\Gamma	Γ	\Pi	Π	\Theta	Θ	\Xi	Ξ
\iota	ι	\psi	ψ	\upsilon	υ	\zeta	ζ

Kalligrafische Buchstaben stehen nur als Großbuchstaben zur Verfügung. Zur Darstellung dieser Buchstaben müssen Sie mit `\cal` die Schriftart verändern, also beispielsweise `{\cal A} \cup {\cal B}` für $\mathcal{A} \cup \mathcal{B}$ schreiben.

8.6 Steuerung des Layouts

Im Großen und Ganzen führt LaTeX den Zeilen- und Seitenumbruch selbstständig durch und kommt dabei zu recht guten Ergebnissen. Es gibt aber auch Situationen, in denen LaTeX versagt. Fallweise (aber nicht immer) zeigt LaTeX Probleme beim Umbruch durch Warnungen der Form *over-/underfull hbox/vbox* an. Damit ist gemeint, dass LaTeX Text außerhalb des Seitenrands platziert hat oder dass im Text sehr große Abstände zwischen den Wörtern auftreten. Die wahrscheinlichste Fehlerursache ist ein langes Wort, in dem LaTeX keine Trennmöglichkeit erkannt hat (speziell bei Fachausdrücken).

Aber auch wenn LaTeX keine Warnungen liefert, kann es vorkommen, dass Sie mit dem Ergebnis nicht zufrieden sind – etwa weil LaTeX versucht hat, eine Seite optimal auszunutzen und dabei eine oder zwei Zeilen eines neuen Absatzes noch auf dieser Seite platziert hat, obwohl der Beginn einer neuen Seite der Übersichtlichkeit halber sinnvoller wäre. Dieser Abschnitt fasst die wichtigsten Möglichkeiten zusammen, manuell (also durch zusätzliche Kommandos) in den Umbruch einzugreifen.

8.6.1 Trennungen

LaTeX trennt prinzipiell automatisch. Die dabei angewandten Trennregeln sind überraschend zuverlässig. Das heißt, es kommt nur sehr selten vor, dass LaTeX wirklich falsch trennt. Sehr viel öfter tritt der Fall ein, dass LaTeX keine eindeutige Trennposition entdeckt, das Wort daher ungetrennt lässt und dann beim Zeilenumbruch in Schwierigkeiten gerät (große Löcher oder Text, der über den Seitenrand hinausragt). In diesem Fall können Sie in das betreffende Wort mit `\-` so genannte *weiche Trennungen* einfügen. LaTeX trennt das Wort dann

– falls notwendig – an einer der von Ihnen angegebenen Stellen (aber dann auch nur an diesen Stellen). Am häufigsten treten Trennprobleme bei Wörtern mit deutschen Sonderzeichen auf, beispielsweise bei `an\-schlie\-ßend` oder `Schlüs\-sel\-wör\-ter`.

Manchmal tritt auch das umgekehrte Problem auf – Sie möchten vermeiden, dass LaTeX ein (oft kurzes) Wort trennt. Dazu stellen Sie einfach das gesamte Wort in ein `\mbox{}`-Kommando.

Trennung beeinflussen	
`\-`	weiche Trennung
`\mbox{wort}`	Wort nicht trennen

In den LaTeX-Distributionen NTeX und teTeX sind bereits die Trenndateien für Englisch, Amerikanisch, Deutsch und einige andere Sprachen eingebaut. Die deutschen Trennungen werden durch das `german`-Package aktiviert. Wenn Ihr Text gemäß der neuen Rechtschreibung getrennt werden soll, müssen Sie statt `german` das Paket `ngerman` verwenden.

Mithilfe des Befehls `\hyphenation{Donau-dampf-schiff}` kann LaTeX mitgeteilt werden, dass das Wort (hier Donaudampfschiff) generell an den mit Minuszeichen gekennzeichneten Stellen getrennt werden darf. Es kann nicht nur ein Wort angegeben werden, sondern durch Leerzeichen getrennt eine ganze Liste von Trennvorschlägen. Leider dürfen in der Trennliste keine Wörter mit deutschen Umlauten vorkommen und die Liste darf auch nicht zu lang werden (max. 300 Wörter). Somit ist die `\hyphenation`-Anweisung nur für die wichtigsten Tennungen zu verwenden. Sie sollte auch für jedes Dokument neu erstellt bzw. angepasst werden.

8.6.2 Wortzwischenräume und horizontale Leerräume

Um die einzelnen Zeilen eines Absatzes im Blocksatz darzustellen, fügt LaTeX zwischen allen Wörtern einen (innerhalb einer Zeile) einheitlichen Leerraum ein und nach einem Punkt (Satzende) einen etwas größeren. Wenn nach einem Punkt nur ein normaler Abstand verwendet werden soll (etwa bei Abkürzungen), muss nach dem Punkt `\␣` angegeben werden: `Dr.\ Huber` für Dr. Huber. (␣ steht für ein Leerzeichen.) Wenn zwei Wörter nicht durch einen Zeilenumbruch getrennt werden sollen, kann ein fixes Leerzeichen mit `~` angegeben werden, beispielsweise `3~cm`.

Zusätzlicher Leerraum zwischen zwei Wörtern kann durch `\quad`, `\qquad` oder `\hspace{abstand}` eingefügt werden. Syntaktisch überflüssige Leerzeichen vor bzw. nach diesen Kommandos sollten vermieden werden, weil sich dadurch ungewollt ein größerer Leerraum ergeben kann.

Die drei oben genannten Kommandos erzeugen Abstände einer genau vorgegebenen Breite. Ganz anders sieht die Wirkung von `\hfill` aus. Dieses Kommando

fügt den gesamten zur Verfügung stehenden Freiraum einer Zeile an der aktuellen Position ein. eigentlich ist `\hfill` eine Abkürzung für `\hspace{\fill}`, wobei `\fill` als „Gummilänge“ bezeichnet wird – eben weil sie variabel ist. Wird `\hfill` in einer Zeile mehrfach verwendet, verkleinert sich der eingefügte Raum entsprechend. Wenn `\hfill` am Ende einer Zeile verwendet wird, muss die Zeile mit `\hbox{}` abgeschlossen werden. Damit wird ein unsichtbares LaTeX-Objekt erzeugt, das als Grenze für `\hfill` wirkt. Beispiel:

```
\hfill zentriert \hfill\hbox{}
```

zentriert

Zusätzliche vertikale und horizontale Abstände	
`\␣`	Wortzwischenraum nach Interpunktionszeichen
`~`	fixes Leerzeichen; an dieser Stelle erfolgt kein Zeilenumbruch
`\quad`	zusätzlicher Leerraum der Größe 1 em (siehe unten)
`\qquad`	zusätzlicher Leerraum von 2 em (siehe unten)
`\hspace{abstand}`	Leerraum der angegebenen Größe einfügen
`\hspace*{abstand}`	wie oben, aber auch bei Zeilenumbruch
`\hfill`	so großen Abstand einfügen, dass Zeile ausgefüllt wird
`\dotfill`	wie oben, aber statt Leerraum punktierte Linie
`\hrulefill`	wie oben, aber durchgezogene Linie
`\hbox{}`	unsichtbares LaTeX-Objekt (als Abgrenzung für `\hfill`)

8.6.3 Zeilenumbruch und vertikale Leerräume

Innerhalb eines Absatzes beginnt LaTeX generell nur dann eine neue Zeile, wenn in der aktuellen Zeile kein Platz mehr ist. Einen vorzeitigen Zeilenwechsel können Sie mit `\\` erreichen. Wenn Sie dahinter in eckigen Klammern einen Abstand angeben, fügt LaTeX außerdem einen entsprechenden vertikalen Leerraum ein: `\\[1cm]`. Wenn zwischen `\\` und der Maßangabe ein `*` steht, wird der Leerraum auch dann eingefügt, wenn ein Seitenwechsel durchgeführt wird. (Das ist nur in den seltensten Fällen sinnvoll!) Zwischen zwei Absätzen kann mit `\vspace{abstand}` ein zusätzlicher Abstand eingefügt werden. Für alle Maße sind auch negative Zahlen erlaubt. Auch bei `\vspace` kann man die Gummilänge `\fill` verwenden. `\vspace{\fill}` oder kurz `\vfill` füllt die Seite bis zum nächsten `\newpage`-Befehl. Man kann so beispielsweise Seiten erzeugen, die nur oben und unten etwas Text enthalten und dazwischen leer sind.

Manueller Zeilenumbruch

`\\`	Zeilenwechsel ohne Randausgleich (kein Blocksatz in dieser Zeile)
`\\[abstand]`	Zeilenwechsel mit erhöhtem Abstand zur nächsten Zeile
`\\*[abstand]`	wie oben, aber Abstand auch bei Seitenwechsel
	Zeilenwechsel mit Randausgleich (Blocksatz)
`\vspace{abstand}`	zusätzlicher Abstand zwischen zwei Absätzen
`\vspace*{abstand}`	wie oben, aber Abstand auch bei Seitenwechsel

8.6.4 Fester Seitenumbruch

Ein fester Seitenumbruch kann mit den drei Kommandos `\newpage`, `\pagebreak` und `\clearpage` erreicht werden. Die Unterschiede gehen aus der folgenden Tabelle hervor.

Manueller Seitenumbruch	
`\newpage`	beginnt eine neue Seite/Spalte, der Rest der Seite bleibt leer
`\pagebreak`	wie oben mit vertikalem Randausgleich (größerer Absatzabstand)
`\clearpage`	neue Seite, auch bei zweispaltigem Text

8.6.5 Eigene Kopfzeilen

Normalerweise kümmert sich LaTeX selbstständig um Kopfzeilen und gestaltet diese je nach Texttyp. Am aufwändigsten geht LaTeX dabei beim Texttyp *book* vor. Es unterscheidet zwischen geraden und ungeraden Seiten und nimmt die Kapitel- und Abschnittsüberschrift mit in die Kopfzeilen auf. Man nennt dies „lebende Kolumnentitel“. Wenn Sie mit den Kopf- und Fußzeilen nicht zufrieden sind, können Sie über `\pagestyle` die automatischen Kopfzeilen deaktivieren und via `\markright` oder `\markboth` eigene Kopfzeilen definieren. LaTeX fügt den so definierten Kopfzeilen automatisch die laufende Seitenzahl hinzu.

Kopfzeilen	
`\pagestyle{headings}`	automatische Gestaltung der Kopfzeile (Default-Einstellung)
`\pagestyle{empty}`	keine Kopfzeile, keine Seitenzahl
`\pagestyle{plain}`	keine Kopfzeile, Seitenzahl zentriert in der Fußzeile
`\pagestyle{myheadings}`	eigene Kopfzeile, siehe `\markright` und `\markboth`
`\thispagestyle`	wie `\pagestyle`, aber nur für eine Seite
`\markright{Kopfzeile}`	Kopfzeile für einseitige Texte
`\markboth{links}{rechts}`	Kopfzeile für zweiseitige Texte

Mit den hier beschriebenen Kommandos ist es nicht möglich, die optische Gestaltung der automatischen LaTeX-Kopfzeilen zu verändern. LaTeX stellt die Überschriften in den Kopfzeilen standardmäßig in Großbuchstaben und ohne Unterstreichung dar, also ganz anders als in diesem Buch. Wenn ein anderes Layout gewünscht wird, muss die Einstellung durch eine eigene Style-Datei verändert werden. Noch eleganter geht es, wenn die Style-Datei `fancyhdr.sty` zur Verfügung steht (`{\usepackage{fancyhdr}`). Ein Beispiel, wie man die LaTeX-typischen Kopfzeilen los wird:

```
...
\pagestyle{fancy}
\renewcommand{\chaptermark}[1]{\markboth{\thechapter\ #1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{}
\fancyhead[LE,RO]{\thepage}
\fancyhead[LO]{\nouppercase \rightmark}
\fancyhead[RE]{\nouppercase \leftmark}
...
```

Damit sind die Kopfzeilen klein und die Seitennummer steht immer ganz aussen. Das ist schlicht und einfach, sieht aber gut aus.

8.6.6 Globale Layouteinstellung

Das Layout eines LaTeX-Textes kann nicht nur durch diverse, zumeist nur auf kleine Textausschnitte angewandte Kommandos beeinflusst werden, sondern auch durch globale Einstellungen. Diese Einstellungen werden normalerweise vor dem eigentlichen Beginn des Textes (also vor `\begin{document}`) vorgenommen und gelten dann für den gesamten Text. Viele der im Folgenden aufgelisteten Kommandos können aber auch irgendwo im Text verwendet werden und entfalten ihre Wirkung dann erst ab dieser Stelle.

Zur Veränderung der meisten Maßangaben existieren eigene Kommandos. Dabei gilt folgende Syntax: `\setlength\kommando{`*maß*`}`. Einige Einstellungen müssen durch eine direkte Veränderung von LaTeX-Variablen durchgeführt werden. In diesem Fall lautet die Syntax: `\setcounter{name}{wert}`.

Die folgende Tabelle fasst die Maße zur Einstellung der bedruckten Bereiche einer Seite zusammen. Die Kommandos sind dabei logisch von links nach rechts bzw. von oben nach unten geordnet.

Seitenlayout

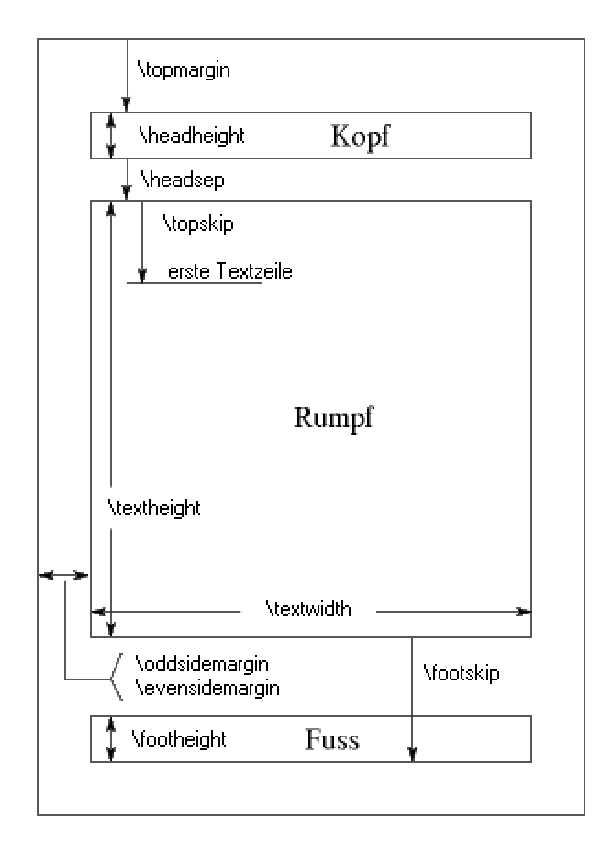


Abbildung 8.6: Seiteneinteilung eines LaTeX-Dokuments

`\setlength{\oddsidemargin}{` *maß* `}`	Abstand linker Papierrand zum Text (ungerade Seiten)
`\setlength{\evensidemargin}{` *maß* `}`	Abstand linker Papierrand zum Text (gerade Seiten)
`\setlength{\textwidth}{` *maß* `}`	Textbreite (bedruckter Bereich)
`\setlength{\topmargin}{` *maß* `}`	Abstand oberer Papierrand zur Kopfzeile
`\setlength{\headheight}{` *maß* `}`	Höhe der Kopfzeile
`\setlength{\headsep}{` *maß* `}`	Abstand Kopfzeile – Text
`\setlength{\textheight}{` *maß* `}`	Texthöhe (für den eigentlichen Text ohne Kopf- und Fußzeilen)
`\setlength{\columnsep}{` *maß* `}`	Spaltenabstand (nur bei zweispaltigem Text)
`\setlength{\columnseprule}{` *maß* `}`	Stärke der Linie zwischen den Spalten (default 0)
`\setlength{\footskip}{` *maß* `}`	Abstand zwischen Text und dem unteren (!) Ende der Fußzeile
`\setlength{\footheight}{` *maß* `}`	Höhe der Fußzeile

Standardmäßig fügt LaTeX zwischen zwei Absätzen keinen Zwischenraum ein. Dafür wird ab dem zweiten Absatz eines Abschnitts die jeweils erste Zeile des Absatzes ein wenig eingerückt. Wenn Sie eine Absatzgestaltung wie in diesem Buch erreichen möchten (kein Einzug, dafür spürbarer Absatzabstand), müssen Sie `\parindent` auf 0 setzen (Maßeinheit nicht vergessen – trotz der 0).

Normalerweise macht LaTeX keinen zusätzlichen vertikalen Abstand zwischen den einzelnen Absätzen. Man kann aber durch Aufruf der Befehle `\smallskip`, `\medskip` oder `\bigskip` dafür sorgen, einen kleinen vertikalen Abstand (viertel, halbe oder ganze Zeile) zwischen allen Absätzen einzufügen. Insofern gehören die Befehle in die Style-Datei oder in den Dokumentenvorspann. Ein Abstand beliebiger Höhe kann durch das Kommando `\renewcommand{\parskip{höhe}}` festgelegt werden.

Erklärungsbedürftig ist auch das Kommando `\flushbuttom`: Es bewirkt, dass LaTeX einen vertikalen Randausgleich durchführt. Dabei wird zwischen Absätzen und Überschriften so viel Leerraum eingefügt, dass alle Seiten exakt gleich lang sind. Standardmäßig ist diese Formatierung nur beim Texttyp *book* aktiviert und kann dort mit `\raggedbottom` abgeschaltet werden.

Absatzlayout

`\parindent`*maß*	Einzug der ersten Zeile des Absatzes
`\bigskip`	einzeiliger Abstand zwischen Absätzen
`\medskip`	halbzeiliger Abstand zwischen Absätzen
`\smallskip`	viertelzeiliger Abstand zwischen Absätzen
`\parskip`*maß*	Abstand zwischen zwei Absätzen
`\raggedright`	kein Blocksatz, sondern Flattersatz
`\flushbottom`	vertikaler Randausgleich (gleich bleibende Seitenhöhe)
`\raggedbottom`	kein vertikaler Randausgleich

Die Syntax zur Veränderung des Zeilenabstands weicht von der der obigen Kommandos ab: Mit `\renewcommand{\baselinestretch`*faktor*`}` wird der normale Zeilenabstand um den angegebenen Faktor vergrößert oder verkleinert. Diese etwas ungewöhnliche Definition ist notwendig, weil der Zeilenabstand von der aktuellen Schriftgröße abhängig ist und daher nicht auf einen starren Wert gesetzt werden sollte.

Üblicherweise nummeriert LaTeX alle Überschriften bis einschließlich `\subsubsection` automatisch durch, was zu Abschnittsnummern wie 3.5.2.1 führt. In diesem Buch wurde die Nummerierung durch eine Veränderung der Variablen *secnumdepth* auf zwei Ebenen (Kapitel und Abschnitte) reduziert. Analog kann die Anzahl der Ebenen für das Inhaltsverzeichnis über die Variable *tocdepth* gesteuert werden (in diesem Buch drei Ebenen).

Nummerierung von Überschriften

`\setcounter{secnumdepth}{n}`	n+1 Gliederungsebenen nummerieren
`\setcounter{tocdepth}{n}`	n+1 Gliederungsebenen ins Inhaltsverzeichnis
`\setcounter{page}{n}`	verändert die laufende Seitenzahl
`\setcounter{chapter}{n}`	verändert die laufende Kapitelzahl
`\setcounter{section}{n}`	verändert die laufende Abschnittszahl

8.7 Briefe schreiben

Da haben Sie sich bei diversen Projekten mit LaTeX angefreundet und nun wollen Sie einen Brief schreiben – und Sie überlegen, wie lange das wohl dauert, bis Sie das mit allen Feldern (Anschrift, Absender, Anlagen, Faltmarken usw.) hingefrickelt haben. Vielleicht hat Sie ein Versuch mit der Umgebung `letter` auch nicht so ganz befriedigt. Für LaTeX existiert glücklicherweise die `dinbrief`-Umgebung, die recht brauchbar ist. Sie macht z. B. standardkonforme Briefe mit Markierungen zum Falten am linken Rand und der Empfängeradresse genau an der Stelle für Fensterumschläge usw.

Ein Dokument kann mehrere Briefe enthalten, die jeweils innerhalb einer `dinbrief`-Umgebung angegeben werden. Eine entscheidende Bedeutung beim Schreiben von Briefen kommt dem `\opening`-Kommando zu: nur dieser Befehl setzt den Briefkopf, die Absenderangaben und die Empfängeradresse. Darauf folgt der eigentliche Brieftext. Abschließend steht der `\closing`-Befehl, der mit zusätzlichen optionalen Argumenten eine Unterschrift als Text oder Grafik einbindet.

Mithilfe der Dokumentenklasse *dinbrief* oder der entsprechenden KOMA-Script-Variante kann man auf einfache Art und Weise einen DIN-gerechten Brief verfassen. Wer zu faul zum Lesen der Dokumentation ist, findet hier eine passende Vorlage. Die einzelnen Elemente erklären sich nahezu von selbst:

```
\documentclass[norm]{dinbrief}
\usepackage{german}
\usepackage[latin1]{inputenc}

\begin{document}
\pagestyle{empty}          % keine Seitennummern sonst {plain}

%
\Absender{\textbf{Hans Mustermann\\
          Musterstr. 7\\[2ex]
          12345 Musterhausen}}
%
% Adresse etwas über die Linie heben
\backaddress {\raisebox{0.6mm}{Hans Mustermann
               $\bullet$ Musterstr. 7
               $\bullet$ 12345 Musterhausen}}
```

```
%
\signature{Hans Mustermann}
%
\Datum{\today}
\place{Musterhausen}
%
%--- Empfänger
\begin{letter}{Name\\
              Straße \par
              PLZ Ort}
%
%--- Betreff
\subject {\textbf{Betreff}}
%
\nowinwowrules % keine Linien um die Adresse
%--- Anrede
\opening{Sehr geehrte Damen und Herren,}

%--- Brieftext


Hier der Brieftext\\[3cm]

blabla bla blabla bla bla\\
blabla bla blabla bla bla\\[3cm]


%
%--- Briefende
\closing{Mit freundlichen Grüßen}
%
%--- Anlagen
% --> ggf. löschen/auskommentieren
\encl{Diverse Anlagen}
%
\end{letter}
\end{document}
```

Die Befehlsliste für den Brief ist nicht sehr umfangreich und schnell zu lernen. Wenn man sich dann noch ein Shellskript schreibt, das den Lieblingseditor mit einer Kopie der Briefvorlage aufruft und auch gleich die Übersetzung und den Druck erledigt, geht so ein Brief auch mit LaTeX ganz fix.

dinbrief-Befehle

Befehl	**Bedeutung**
`\address{}`	Name und Adresse des Absenders
`\stdaddress{}`	Absenderadresse nach DIN 5008
`\signature{}`	Unterschrift des Absenders
`\backaddress{}`	Absenderadresse im Brieffenster
`\place{}`	Ortsangabe im Brief (Dingenskirchen, den ...)
`\date{}`	Briefdatum
`\yourmail{}`	Ihre Zeichen, Ihre Nachricht vom
`\sign{}`	Unsere Zeichen (, unsere Nachricht vom)
`\phone{}`	Rufnummer
`\writer{}`	Sachbearbeiter
`\subject{}`	1. Betreff
`\concern{}`	2. Betreff
`\opening{}`	Anrede
`\closing{}`	Grußformel
`\encl{}`	Anlagen
`\ps{}`	Postscriptum
`\cc{}`	Verteiler
`\bottomtext{}`	Konto, Handelsregister usw.
`\postremark{}`	Postvermerk
`\handling{}`	Behandlungsvermerk
`\centeraddress`	Absenderadresse zentrieren
`\normaladdress`	Absenderadresse linksbündig
`\nowindowrules`	keine Linien um die Empfängeradresse
`\windowrules`	Linien um die Empfängeradresse
`\nowindowtics`	keine Faltmarken
`\windowtics`	Faltmarken

Markus Kohm, der federführende Autor des KOMA-Script-Pakets, will mit seinem Paket Bedürfnisse einer europäischen Typografie bedienen. Eigens für die Erstellung von Briefen hat Kohm eine Klasse *scrlttr2* geschaffen. Das Paket ersetzt die Vorgängerversion *scrlettr*. Für eine eingehendere Beschäftigung mit *scrlttr2* verweisen wir auf die deutschsprachige Dokumentation (texmf/doc/latex/koma-script/scrguide.pdf).

Es lassen sich auch sehr leicht Serienbriefe schreiben. Sie benötigen dazu nur ein kleines Makro wie das folgende (wie man Makros selbst schreibt, erfahren Sie im folgenden Abschnitt):

```
\newcommand{\Brief}[1]{
  \begin{letter}{#1}
  \input{Brieftext}
  \end{letter}}
```

Mit dem Befehl `\input{Brieftext}` wird der Text des Serienbriefs aus der Datei *Brieftext* geladen, die natürlich wieder alle LaTeX-Elemente von *dinbrief* enthalten kann. Das Makro packen Sie mit in das Dokument:

```
\documentclass[norm]{dinbrief}
\usepackage{german}
\usepackage[latin1]{inputenc}

\newcommand{\Brief}[1]{
  \begin{letter}{#1}
  \input{Brieftext}
  \end{letter}}

\begin{document}
% Vorspann wie oben
  ...
% nur statt \begin{letter} kommen nun die Makroaufrufe
\Brief{Thusnelda Bims\\
       Lange Gasse 123\\
       98765 Sonstwo}
\Brief{Waldemar von Adlig\\
       Königstr. 12\\
       45678 Burghausen}
  ...
%
\end{letter}
\end{document}
```

Auch die Makro-Aufrufe mit den Adressen können Sie in eine zweite Datei auslagern, die dann per `\input`-Befehl gelesen wird. Eine solche Daten kann auch leicht per Programm oder Datenbankabfrage generiert werden.

Auch lässt sich das neu definierte Makro noch ausblasen, um beispielsweise unterschiedliche Anreden (Herr, Frau, ...) zu generieren. Außerdem gibt es fertige LaTeX-Pakete für Serienbriefe, unter anderem „mailing“, „textmerg“, „finder“ und „formletter“.

8.8 Farben

In LaTeX können Sie Farben nach dem RGB (rot grün blau)-Farbmodell (additive Farbmischung, z. B. beim Monitor), CMY(K) (cyan magenta yellow (black))-Farbmodell (subtraktive Farbmischung, z. B. beim Drucker) oder HSB (hue saturation brightness) sowie in Graustufen definieren. Es wird das Paket *color* benötigt.

LaTeX-Farbmodelle

Name	*Basisfarben*	*Wertebereich*
rgb	red, green, blue	0.0 ... 1.0
cmy	cyan, magenta, yellow	0.0 ... 1.0
cmyk	cyan, magenta, yellow, black	0.0 ... 1.0
hsb	hue, saturation, brightness	0.0 ... 1.0
gray	gray	0.0 ... 1.0
RGB	Red, Green, Blue	0 ... 255
HSB	Hue, Saturation, Brightness	0 ... 255
Gray	gray	0 ... 15

Bei *RGB, HSB* und *Gray* sind nur ganze Zahlen erlaubt, bei allen anderen Modellen die üblichen Zahlen mit Kommastellen. Außerdem gibt es noch das Modell *named* mit vordefinierten Farben (s. u.), die aber nicht jeder DVI-Treiber akzeptiert.

TeX-Farben sind in der Regel nach dem CMYK-Modell definiert, weil der Drucker das typische Ausgabegerät ist. LaTeX greift auf die Datei `color.pro` zu, in der diese Definitionen stehen, außerdem wird `color.pro` ggf. von DVIPS-Treibern gelesen (Wobei die Linux-Version von `xdvi` keine Farben darstellt, Sie müssen die DVI-Datei erst in eine PS-Datei umwandeln, um die Farben zu sehen.) Diese Farben treffen – bis auf wenige Ausnahmen – annähernd Farbtöne aus der PANTONE-Palette. PANTONE ist der Standard in der Druckindustrie. Die Farbauszeichnung nach dem PMS (Pantone Matching System) sollte daher auf verschiedenen Druckern ein in etwa gleiches Druckergebnis gewährleisten.

Bei Folien (siehe nächsten Abschnitt) liegt der Fall etwas anders, hier kann man sich auf das RGB-Modell stützen, bei dem die Farben aus den Anteilen Rot, Grün und Blau gemischt werden – wie beim Fernsehen oder auch bei den Farbangaben von HTML-Seiten. Die Definition der Farben erfolgt mit dem Befehl `\definecolor{Farbname}{Modell}{Werte}`, die Umschaltung auf die Farbe dann mit `\color{Farbname}`. Das Verhalten ist hier wie beim Fettdruck oder der Schriftgröße – entweder man schaltet die Farbe dauerhaft jeweils um oder man schließt den einzufärbenden Bereich in geschweifte Klammern ein. Beispiele für Farbdefinitionen:

```
% Modell RGB, Werte für Rot-, Grün- und Blau-Anteil, durch Komman getrennt
\definecolor{white}{rgb}{1,1,1}
\definecolor{black}{rgb}{0,0,0}
\definecolor{gold}{rgb}{1.0,0.84,0}
\definecolor{pastellgruen}{rgb}{0.855,1.0,0.90}
\definecolor{pastellblau}{rgb}{0.85,0.95,1.0}
\definecolor{pastellpink}{rgb}{0.965,0.785,0.89}
\definecolor{pastellgelb}{rgb}{0.875,1.0,0.75}

Hier ist ein Wort in {\color{pastellpink} pastellpink} eingefärbt.
```

Immer verfügbar sind die vordefinierten Farben mit den Namen *black, white, red, green, blue, cyan, magenta* und *yellow.*

Bei der Anwendung von `dvips` werden Sie feststellen, dass nicht alle Farben gleich gut wirken. Manche Pastelltöne kommen zu schwach oder auch zu kräftig. Auch fällt manchem die Namenswahl schwer. Daher sind in der Datei `dvipsnam.def` etliche Farbnamen vordefiniert. Mit folgendem LaTeX-Dokument können Sie sich eine Farbtabelle erzeugen (das neu definierte Kommando `SC` zeigt die jeweilige Farbe als rechteckiges Feld und ihren Namen).

```
\documentclass[12pt,a4paper]{article}
\usepackage[usenames,dvipsnames]{color}
\usepackage{multicol}
\pagestyle{empty}
\setlength{\parindent}{0pt}

\newcommand{\SC}[1]{%
\textcolor[named]{#1}{\rule{10mm}{10mm}}\quad
\texttt{#1}\strut\\}

\begin{document}
\noindent
\begin{multicols}{3}
\SC{GreenYellow}   \SC{Yellow}       \SC{Goldenrod}
\SC{Dandelion}     \SC{Apricot}      \SC{Peach}
\SC{Melon}         \SC{YellowOrange} \SC{Orange}
\SC{BurntOrange}   \SC{Bittersweet}  \SC{RedOrange}
\SC{Mahogany}      \SC{Maroon}       \SC{BrickRed}
\SC{Red}           \SC{OrangeRed}    \SC{RubineRed}
\SC{WildStrawberry}\SC{Salmon}       \SC{CarnationPink}
\SC{Magenta}       \SC{VioletRed}    \SC{Rhodamine}
\SC{Mulberry}      \SC{RedViolet}    \SC{Fuchsia}
\SC{Lavender}      \SC{Thistle}      \SC{Orchid}
\SC{DarkOrchid}    \SC{Purple}       \SC{Plum}
\SC{Violet}        \SC{RoyalPurple}  \SC{BlueViolet}
\SC{Periwinkle}    \SC{CadetBlue}    \SC{CornflowerBlue}
\SC{MidnightBlue}  \SC{NavyBlue}     \SC{RoyalBlue}
\SC{Blue}          \SC{Cerulean}     \SC{Cyan}
\SC{ProcessBlue}   \SC{SkyBlue}      \SC{Turquoise}
\SC{TealBlue}      \SC{Aquamarine}   \SC{BlueGreen}
\SC{Emerald}       \SC{JungleGreen}  \SC{SeaGreen}
\SC{Green}         \SC{ForestGreen}  \SC{PineGreen}
\SC{LimeGreen}     \SC{YellowGreen}  \SC{SpringGreen}
\SC{OliveGreen}    \SC{RawSienna}    \SC{Sepia}
\SC{Brown}         \SC{Tan}          \SC{Gray}
\SC{Black}         \SC{White}
\end{multicols}
\end{document}
```

Um die Farben zu nutzen, gibt es mehrere Möglichkeiten. Neben dem Befehl `\color` gibt es für kurze Textstücke `\textcolor{Farbe}{Text}`, der einzelne Wörter oder Passagen einschließt.

Auf die gleiche Weise lässt sich auch die Hintergrundfarbe des Textes modifizieren. `\pagecolor{Name}` bzw. `\pagecolor [modell] {spezifikation}` setzt die Hintergrundfarbe der Seite. Dieser Befehl wirkt immer auf die gesamte Seite und bis zum Ende des Dokuments. Um eine einzelne Seite farbig zu hinterlegen, eignet sich die Befehlsfolge

```
\pagecolor{yellow}%
\afterpage{\pagecolor{white}}
```

Der Befehl `\normalcolor` dient zum Zurücksetzen der Farbeinstellungen auf die Anfangswerte.

Neben der Einfärbung von Schrift und Seitenhintergrund gibt es die Möglichkeit, den Hintergrund einer (Text)Box farbig zu gestalten. Der Befehl `\colorbox{Name}{Text}` hinterlegt den Text mit einer Farbbox ohne Rahmen. Statt des Farbnamens kann auch die Kombination `[Modell]{Spezifikation}` angegeben werden. Die Box hat einen Rand der Breite `\fboxsep`. Wenn Sie einen zusätzlichen Rahmen wollen, verwenden Sie den Befehl `\fcolorbox{R}{H}{Text}`, der wie `\fbox` funktioniert. *R* spezifiziert die Rahmenfarbe (Name) und *H* die Hintergrundfarbe. `\fboxrule` steuert die Rahmenbreite. Dazu ein Beispiel:

```
\colorbox{gray}{Ein grau unterlegter Text}

\setlength{\fboxsep}{5mm}
\setlength{\fboxrule}{3mm}
\fcolorbox{red}{blue}{Ein nichtssagender Text}
```

Das ergibt (hier sind aus drucktechnischen Gründen auch Blau und Rot als Graustufen dargestellt worden):

Ein grau unterlegter Text



Color-Befehle

Befehl	Bedeutung
`\color{Farbe}`	setzt die Textfarbe auf die angegebene Farbe
`\textcolor{Farbe}{Text}`	desgleichen für kurze Textstücke
`\normalcolor`	detzt die Farbe auf den Defaultwert
`\pagecolor{Name}`	einfärben des Seitenhintergrunds
`\colorbox{Name} {Text}`	farbiges Hinterlegen des Textes (So wie bei diesem Kasten)
`\fcolorbox{R}{H}{Text}`	text farbig hinterlegen mit Farbe *H*; zusätzlich ein Rahmen der Farbe *R*

Bei den meisten Befehlen ist statt eines Farbnamens auch die Angabe von Modell und Farbspezifikation (`\befehl[modell]{spezifikation}` möglich.

8.9 Texte rotieren

Manchmal möchte man Texte nicht in der normalen Richtung anordnen, sondern senkrecht oder in irgendeinem Winkel gegenüber der Waagrechten. Da hilft das Paket `rotating`. Es stellt drei neue Umgebungen zur Verfügung, `rotate`, `turn` und `sideways`. Alle bewirken die Rotation des eingefassten Textes um den als Argument angegebenen Winkel. Der Winkel wird als Zahl ohne Einheit angegeben, LaTeX fasst dies als Angabe in (Alt-)Grad auf, gemessen gegen den Uhrzeigersinn von der Horizontalen aus.

- Alles innerhalb der `sideways`-Umgebung wird um 90 Grad entgegen dem Uhrzeigersinn gedreht.

  ```
  \begin{sideways}
    ...
  \end{sideways}
  ```

  Für eingebundene Bilder gibt es `sidewaysfigure`, das anstelle von `figure` verwendet wird. Dazu passend dreht `rotcaption` die Bildunterschrift.

- Alles innerhalb der `rotate`-Umgebung wird um einen bestimmten Winkel gedreht. Dabei wird dem gedrehten Text kein Platz eingeräumt. Man muss daher mit geeigneten Mitteln für genügend Raum sorgen (z. B. mittels Boxen). Es lassen sich damit sehr gut senkrechte Beschriftungen bei Objekten variabler Größe anbringen, wie beispielsweise in den Informationskästen hier im Buch.

  ```
  \begin{rotate}{Winkel}
    ...
  \end{rotate}
  ```

- Bei `turn` wird ausgehend von der linken unteren Ecke im Uhrzeigersinn um `Winkel` Grad gedreht. Anders als bei `rotate` wird dem gedrehten Text genügend Platz gelassen.

```
\begin{turn}{Winkel}
  ...
\end{turn}
```

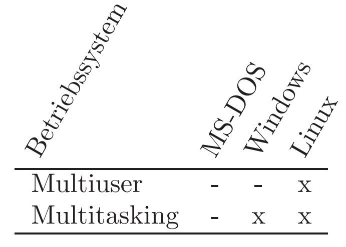
Das folgende Beispiel dient als erste Spielerei:

```
\begin{document}
\begin{turn}{45}\fbox{Um 45° nach oben geneigter Text.}\end{turn}
\begin{turn}{315}\fbox{Um 45° nach unten geneigter Text.}\end{turn}
\begin{turn}{405}\fbox{Um 405° (=360+45°) geneigert Text.}\end{turn}
\begin{turn}{180}\fbox{Extra für die Australier.}\end{turn}
\end{document}
```

Gut geeignet ist die Rotation, wenn man schmale Tabellen setzen will, die aber relativ lange Spaltenlegenden haben. Dazu ein Beispiel:

```
\begin{tabular}{lccc}
\rule{0mm}{20mm} % Platz für Legende frei halten
\begin{rotate}{60}Betriebssystem\end{rotate}\hfill &
\begin{rotate}{60}MS-DOS\end{rotate} &
\begin{rotate}{60}Windows\end{rotate} &
\begin{rotate}{60}Linux\end{rotate} \\
\hline
Multiuser & - & - & x \\
Multitasking & - & x & x \\
\hline
\end{tabular}
```

Das Beispiel liefert als Ergebnis:



Betriebssystem	MS-DOS	Windows	Linux
Multiuser	-	-	x
Multitasking	-	x	x

8.10 Folien und Präsentationen erstellen

Für Präsentationen wird heuzutage in der Regel Powerpoint verwendet und man kann schon froh sein, wenn der Vortragende dabei gewisse Grundregeln einhält, etwa des Verwenden großer Schrift und das Darbieten von nicht zuviel Informationen. Steht dem Vortragenden kein Beamer zur Verfügung, werden die PP-Seiten einfach auf Folien gedruckt. Dabei gilt oft: Form dominiert über Funktion – d. h. Logos, Rahmen, Farbverläufe usw. lassen die Informationen beinahe nicht mehr erkennen. Dafür dauert es Stunden, bis die Folie fertig ist.

Sie können aber auch LaTeX verwenden, um Folien zu erstellen, und dabei unter etlichen, teilweise recht mächtigen Paketen wählen. Wir wollen hier zwei Pakete vorstellen: `seminar`, ein einfaches Paket zum Erstellen von Vortragsfolien, das man schon als „Großvater aller Präsentationen“ bezeichen könnte, und einen Abkömmling, `beamer`, der als PDF-Präsentation viele interaktive Möglichkeiten bietet.

8.10.1 Folien erstellen mit Seminar

Das Paket ist in der Regel schon in der LaTeX-Installation enthalten. Falls nicht, erhalten Sie es unter *http://www.tug.org/applications/Seminar/*. Das Paket wurde von Timothy Van Zandt entwickelt und zeichnet sich wie gesagt durch ein einfaches Layout aus. Es bietet nur wenige Features, z. B. fehlt das schrittweise Einblenden von Informationen.

Das Seminar-Paket unterscheidet zwei Formate von Folien, das Querformat (landscape) und das Hochformat (portrait). Als Grundstil dient die Dokumentenklasse *seminar*. Einzelne Folien werden im normalen Text mit `\begin{slide}` und `\end{slide}` umschlossen. Es werden automatisch größere Schriften etc. verwendet. Der Quellcode für eine super-einfache Landscape-Folie lautet:

```
\documentclass[11pt]{seminar}
% Fontgroessen-Voreinstellung 11pt; möglich sind 11pt oder 12pt
\usepackage{german}
\usepackage[latin1]{inputenc}
\begin{document}
\begin{slide}
Eine Folie im Landscape-Format.
\end{slide}
\end{document}
```

Die Fontgrößen-Angabe entspricht dem, was man bei `article` oder `book` verwenden würde, die real verwendeten Fonts sind jedoch größer). Beim Betrachten mit GhostScript kommt keine Freude auf, es scheint so, als ginge irgendetwas schief – das Papierformat ist im Portrait-Modus, die Folie im Landscape-Modus. Beim Drucken im Landscape-Modus kommen die Folien aber richtig aus dem Gerät. Dem Manko können Sie mit einem kleinen Makro (siehe folgenden Abschnitt) abhelfen. Die Header müssen dann natürlich mitrotiert werden:

```
% folgender Befehl richtet sich nicht an LaTeX,
% sondern wird durchgereicht an dvips
% Seitenorientierung richtig einstellen
\renewcommand{\printlandscape}{\special{landscape}}
\rotateheaderstrue
```

Innerhalb einer `slide`-Umgebung kann mit `\newslide` eine neue Folie erzwungen werden. Normalerweise wird nach `\end{slide}` automatisch eine neue

Folie begonnen. Mit dem Befehl `\extraslideheight{len}` kann der Seitenwechsel gesteuert werden. Wenn Sie beispielsweise `\extraslideheight{100cm}` verwenden, müssen Sie den Seitenumbruch mit `\newslide` erzwingen. `\extraslideheight{0pt}` überlässt LaTeX den Umbruch ohne Kompromisse. Die Voreinstellung ist `\extraslideheight{10pt}`.

Orientierung

Verwendet man `slide*` anstelle von `slide`, wird die Orientierung gewechselt (portait statt landscape). Man kann auch durch die Option `portrait` bei `\documentclass` gleich das Hochformat als Standard einstellen (auch dann muss `slide*` für Folien im Hochformat verwendet werden). Durch die Befehle `landscapeonly` und `portraitonly` kann man die Ausgabe so steuern, dass jeweils nur Folien der entsprechenden Orientierung ausgegeben werden.

Das Seminar-Paket bietet den Befehl `\slidesmag{n}` an, um den Folieninhalt zu vergrößern oder verkleinern. Die Größe wird als 1.2^n berechnet. n muss dabei eine ganze Zahl zwischen -5 und 9 sein.

Das Format der Folie kann mit den drei folgenden Befehlen verändert werden:

- `\slidewidth{breite}` gibt die Breite der Folien an.
- `\slideheight{höhe}` gibt die Höhe der Folien an.
- Die Parameter `Breite` und `Höhe` bei `\begin{slide}[Breite,Höhe]` erlauben bei jeder Folie eine Änderung der Größe.
- `\centerslidestrue` (Voreinstellung) zentriert die Folie auf dem Papier, mit `\centerslidesfalse` wird nicht zentriert.
- `\raggedslides[len]` erlaubt es, den Randausgleich am rechten Rand zu steuern. Normalerweise erfolgt Flattersatz (Voreinstellung). Der Wert *len* steuert den Ausgleich, mit `\raggedslides[opt]` bekommen Sie Randausgleich.

Natürlich lassen sich wie bei anderen Dokumentenstilen die Ränder mittels `\slideleftmargin`, `\sliderightmargin`, `\slidetopmargin`, `\slidebottommargin`, `\paperwidth` und `\paperheight` individuell einstellen (siehe Dokumentation zum Paket).

Schriftarten

Die Standardschrift mit Serifen eignet sich wunderbar für Bücher und Artikel, aber weniger für Folien, wo eine serifenlose Schrift besser und klarer zur Geltung kommt. Durch die Option `semhelv` beim `\documentstyle` kann auf die serifenlose Helvetica-Schrift umgeschaltet werden (ggf. ist noch ein

`\usepackage{semhelv}` im Vorspann nötig). Die Schriften im Mathematik-Modus bleiben, wie sie waren. Alternativ kann auch mit der Option `semlcmss` die serifenlose Schrift aus den Computer Modern FOnts eingebunden werden.

Ränder und Rahmen

Der Befehl `\slideframe[options]{style}` ermöglicht es, die Ränder der Folien zu gestalten. Mögliche Stile sind `none` oder `plain`. Die `pagestyles` `none`, `plain`, `headings` oder `myheadings` verhalten sich genauso wie bei `article`. Mit den beiden Zeilen

```
\pagestyle{headings}
\markright{Folien mit \textbf{seminar.cls}}
```

erhalten Sie beispielsweise eine Überschrift („Folien ...“) mit Seitennummer auf jeder Folie. Sollen Kopf- und Fußzeilen individuell gestaltet werden, hilft ein eigener Seitenstil, z. B.:

```
\newpagestyle{meiner}%
   {Folien mit \textbf{seminar.cls} \hfill \rightmark \hfill \thepage}%
   {Michael Kofler, Jürgen Plate \hfill  \today}%
\pagestyle{meiner}
```

Hier erscheinen die Überschrift links oben, die Seitennummer rechts oben, die Autoren links unten und das Datum rechts unten. Sie könnten natürlich auch noch ein kleines Logo einbinden oder das Ganze noch aufpeppen. mit `\renewpagestyle` kann der Seitenstil auch jederzeit umdefiniert werden.

Weitere Stile können beispielsweise mit `\documentstyle [fancybox]{seminar}` eingebunden werden, darunter *shadow* (Folienumrandung mit Schatten), *double* (doppelte Folienumrandung), *oval* (ovale Folienumrandung) oder *Oval* (dickere ovale Folienumrandung). Es muss dann aber das Paket *fancybox* eingebunden werden. Zum Beispiel:

```
\documentstyle[12pt,fancybox,semhelv]{seminar}
\usepackage{german}
\usepackage[latin1]{inputenc}
\usepackage{fancybox}
\renewcommand{\printlandscape}{\special{landscape}}
\slideframewidth 0.5cm
\slideframe{oval}
\begin{document}
\begin{slide}
Folie mit ovalem Rand.
\end{slide}
\end{document}
```

`\slideframewidth` legt die Dicke der Rahmenlinie fest (Voreinstellung 4pt) und `\slideframesep` den Abstand zwischen Rahmen und Inhalt (Voreinstellung 0.4in).

Übrigens können Sie natürlich auch die zugehörigen Boxen `\shadowbox{Text}`, `\ovalbox{Text}`, `\Ovalbox{Text}` und `\doublebox{Text}` zur Gestaltung Ihrer Folien verwenden.

Farben

Folien nur in Schwarz und Weiss sind sicher etwas langweilig. Farbe bringt Aufmerksamkeit und kann zur Gliederung des Dargebotenen beitragen. Neben den schon auf Seite 399 behandelten allgemeinen Möglichkeiten zur Einfärbung des Textes werden bei der Seminar-Dokumentenklasse mit `\usepackage{semcolor}` einfache Voraussetzungen für Farben geschaffen. Sie können auch hier beliebige eigene Farben definieren, wobei Ihnen alle Farbmodelle zur Verfügung stehen. Wir bleiben mal bei nur einem Modell, das den meisten am geläufigsten sein dürfte, dem RGB-Modell. Zum Definieren von Farben dient hier der Befehl `\newrgbcolor{Name}{Rotanteil Grünanteil Blauanteil}`, z. B.:

```
\newrgbcolor{gray}{0.9 0.9 0.9}
\newrgbcolor{lred}{1 0.5 0.5}
\newrgbcolor{orange}{1.0 0.5 0.0}
\newrgbcolor{blue}{0.0 0.0 1.0}
```

Damit kann man dann loslegen; die Farbnamen definieren die Textfarbe, mit geschweiften Klammern wird der Gültigkeitsbereich festgelegt. Mittels `\colorbox{Farbe}{Inhalt}` können Sie auch farbig hinterlegte Texte erzeugen:

```
{\gray Dies ist} {\lred eine Folie}
{\orange mit bunten} {\blue Farben}.
...
\colorbox{gray}{Ein grau unterlegter Text}
```

8.10.2 Folien erstellen mit Beamer

Auch dies Paket ist in der Regel schon in der LaTeX-Installation enthalten und bietet alles, was das Herz begehrt (soweit es um das Erstellen von Präsentationen geht). Die Folien sind per default auf die Größe 128 mm x 96 mm, also ein 4:3-Format festgelegt. Das Postkartenformat wird durch den PDF-Betrachter auf Bildschirmgröße skaliert. Es kann bei der Schriftgröße zwischen `11pt` und `12pt` gewählt werden. Navigationselemente werden automatisch erzeugt.

Einige vom Paket benötigten Pakete, darunter *color*, *xcolor* und *hyperref* werden automatisch geladen.

Das Vorgehen beim Produzieren der fertigen Dateien ist etwas anders als bisher (Sie können das übrigens beim Seminar-Paket auch so machen):

- Erstellen der Folien wie unten beschrieben → `folien.tex`
- PDF-Präsentation erstellen: `pdflatex folien.tex` → `folien.pdf`
- Falls Bilder eingebunden werden, müssen diese nicht im PS- oder EPS-Format vorliegen, sondern als JPG, PDF, PNG oder TIFF.
- Einige Pakete erfordern die Angaben von `pdflatex` als Option, beispielsweise `includepackage[pdftex]{color}` oder `includepackage[pdftex]{hyperref}`.

Anstelle des Kommandos `latex folien.tex` erspart man sich hier den Umweg über eine dvi- und eine PostScript-Datei und generiert mittels `pdflatex folien.tex` direkt die PDF-Datei für die Präsentation. Mit dem Acrobat Reader kann man dann die Präsentation im Vollbild-Modus laufen lassen und mit den Cursortasten zwischen den Folien wechseln. Das Programm `pdflatex` ist Bestandteil fast aller LaTeX-Distributionen.

Auch das Beamer-Dokument beginnt mit einigen Standardbefehlen, unter anderem mit der Auswahl der Dokumentenklasse und des Themas bzw. Stils der Präsentation. Beamer bietet zahlreiche Stile, die alle nach Städten benannt sind. Voreingestellt ist `default`. Die Stile heißen:

Beamer-Farbschemata			
Antibes	Bergen	Berkeley	Berlin
Boadilla	Copenhagen	Darmstadt	Dresden
Frankfurt	Goettingen	Hannover	Ilmenau
JuanLesPins	Luebeck	Madrid	Malmoe
Marburg	Montpellier	PaloAlto	Pittsburgh
Rochester	Singapore	Szeged	Warsaw

Bei der Dokumentenklasse kann als optionaler Parameter die Grundfarbe der Präsentation angegeben werden. Wählt man hier eine bestimmte Farbe, z. B. `[red]`, wird die Präsentation in diversen Rottönen gestaltet, bei `[blue]` in Blautönen usw.

Das Grundgerüst eines Beamer-Vortrags stellt sich somit folgendermaßen dar:

```
\documentclass{beamer}
\usepackage{german}
\usepackage[latin1]{inputenc}

\usetheme{Malmoe}   % oder irgend ein anderes

\begin{document}
```

```
\title{\LaTeX\ Beamer Class}
\author{Michael Kofler, Jürgen Plate}
\date{\today}
\maketitle % oder auch \frame{\titlepage}

% Hier ggf. \frame{\tableofcontents}

\begin{frame}
  \frametitle{Einleitung}
  \framesubtitle{Wie Rotkäppchen in den Wald kam}
   ...
\end{frame}

\end{document}
```

Abschnitte

Das Beamer-Interface ist genauso einfach wie das Seminar-Paket. Jede Folie wird in eine `\frame`-Umgebung eingebettet. Außerdem kann auf die `\section`- und `\subsection`-Struktur zurückgegriffen werden (wie bei der Dokumentenklasse `article`). Die Struktur wird in den Kopf der Folien übernommen. Mit dem Befehl `\frame{\tableofcontents}` wird eine Übersichtsseite erzeugt, deren Elemente auch der Navigation dienen. Für mehrteilige Präsentationen, z. B. mehrtägige Schulungen, ist auch eine übergeordnete Gliederung mit dem Befehl `\part` möglich.

Für kurze Frames kann anstelle der `\frame`-Umgebung auch ein Befehl `\frame{...}` verwendet werden. Der Befehl `\verb` ist in Frames nur dann erlaubt, wenn beim `\begin{frame}`-Befehl der optionale Parameter `[containsverbatim]` hinzugefügt wird. Weitere wichtige Optionen sind `[plain]`, welche die Navigationselemente, Kopf- und Fußzeilen unterdrückt. Das wird manchmal benötigt, um seitenfüllend ein Bild zu präsentieren. Als Bezugspunkt für die Navigation kann `[label=name]` eingefügt werden, was eine Wiederholung dieser Folie mit `\againframe` möglich macht.

Blocks und Kästen

Eine weitere Strukturierungsmöglichkeit der Folie bieten abgesetzte Blöcke, die dann je nach gewähltem Layout gestaltet werden (mit/ohne Rahmen, farbig hinterlegt usw.):

```
\begin{frame}
  \frametitle{Titel}
  \begin{block}{Titel des Blocks}
    Hier steht dann der tolle Text zum ersten Block...
  \end{block}
  \begin{block}{Titel des 2. Blocks}
```

```
    Und hier der nicht minder wichtige Text zum 2. Kasten...
  \end{block}
\end{frame}
```

Dabei ist der erste Parameter der *block*-Umgebung die Überschrift des Kastens, die farbig hervorgehoben wird. Auf die gleiche Weise funktioniert der `\alertblock`, bei dem die Überschrift in der vordefinierten Farbe für Alerts (siehe später) hervorgehoben wird.

Ähnliche Umgebungen gibt es für Lehrsätze und alles was dazugehört: `definition`, `example`, `theorem`, `corollary`, `fact`, `proof` und `lemma`.

Ein Frame kann in mehrere Spalten aufgeteilt werden. Der optionale Parameter in eckigen Klammern gibt an, wie die beiden Spalten zueinander vertikal ausgerichtet werden: **b**: nach der untersten Zeile, **c**: zentriert und **t**: nach der obersten Zeile. Man spart sich so bei Gegenüberstellungen das Gefrickel mit `tabbing` oder `tabular` bzw. `minipage`. Beispiel:

```
\begin{columns}[t]
 \begin{column}{2cm}
 Erste Zeile \\
 Zweite Zeile
 \end{column}
 \begin{column}{2cm}
 Nur eine Zeile
 \end{column}
\end{columns}
```

Overlays

Der einfachste Weg, um Overlays zu erzeugen, ist der `\pause`-Befehl. Sie schreiben einfach den `\pause`-Befehl an die Stelle, bis zu der die Folie am Anfang sichtbar sein soll und dann vor jeden weiteren Schritt. Ein Beispiel:

```
\begin{frame}
  \frametitle{Einstieg}
  \pause
  \begin{itemize}
  \item Warum?
      ...
  \pause
  \item Darum!
      ...
  \pause
  \item Zum Schluss:
      ...
   \end{itemize}
\end{frame}
```

Die einzelnen Punkte erscheinen jeweils nach und nach, der Rest der Folie bleibt, wie er ist. Aber das ist nur die einfachste Möglichkeit. Beim Beamer-Paket lassen sich die tollsten Sachen machen. Dazu gibt es spezielle Overlay-Spezifikationen und etliche neue Befehle. Wer jetzt gerade erst mit LaTeX warm geworden ist, wird überrascht sein, denn die Overlay-Spezifikationen werden weder in geschweifte noch in eckige Klammern eingeschlossen, sondern in Kleiner- und Größerzeichen. Also aufpassen! Innerhalb der Klammerung wird festgelegt, auf welchen Folien das jeweilige Element der Folie (genauer: welche LaTeX-Gruppe) erscheinen soll.

Innerhalb einzelner Folien kann man mit den Befehlen `\only` und `\visible` in Kombination mit einer numerischen Angabe eine Animationsreihenfolge erzeugen. Diese Reihenfolge wird durch mehrere Ausgabeseiten pro Folie realisiert, sodass die Präsentationen in jedem PDF-Viewer komfortabel abgespielt werden können. Dazu ein Beispiel:

```
\begin{frame}
  \visible<1>Erste Folie!
  \visible<2->Ab der zweiten Folie.
  \only<3>Nur auf der dritten Folie.
\end{frame}
```

Der Unterschied zwischen `\only` und `\visible` besteht darin, dass `\only` den Text tatsächlich nur auf der Seite erzeugt, für die er vorgesehen ist. `\visible` erzeugt dagegen den Text immer und „druckt“ ihn auf den nichtspezifizierten Seiten mit der Hintergrundfarbe. Daher nehmen `\only`-Elemente auf den übrigen Seiten keinen Platz ein, während `\visible`-Elemente immer Platz beanspruchen (und damit die übrigen Elemente verschieben). In der Regel sorgt letzteres für einen ruhigeren Folienaufbau, da sich der Rest der Darstellung nicht verschiebt.

Overlay-Spezifikationen	
$< n >$	Anzeige nur auf Folie n
$< n- >$	Anzeige ab Folie n
$< -n >$	Anzeige nur bis Folie n
$< n - m >$	Anzeige nur auf Folien n bis m
$< i, j - k, n, m - p >$	Beliebige Kombination der obigen Optionen, durch Kommata getrennt

Eine Angabe wie $< 4- >$ bewirkt also, dass das entsprechende Element auf der vierten und allen folgenden Seiten der jeweiligen Folie sichtbar ist. $< 3 - 5, 8 >$ bedeutet, dass das Element nur auf den Seiten 3, 4, 5 und 8 zu sehen ist. Bei etlichen Konstruktionen von LaTeX ist der `\visible`-Effekt schon eingebaut, z. B. bei Aufzählungen. Um nacheinander Punkte erscheinen zu lassen, geben Sie bei jedem Punkt an, auf welchen Folien er zu sehen ist:

```
\begin{itemize}
```

```
  \item<1-> Erster Punkt, auf jeder Folie.
  \item<2-> Ab der zweiten Folie.
  \item<3-> Ab der dritten Folie.
  \item<4-> Und ab der vierten Folie.
\end{itemize}
```

Die `\item`-Elemente unterstützen also numerische Angaben ohne zusätzliches Kommando. Wenn Sie im Beispiel oben nach dem ersten Punkt einen weiteren einschieben, müssen Sie lästigerweise alle folgenden Optionen ändern. Bei einfachen Aufzählungen wie im Beispiel, bei denen die einzelnen Punkte in der Folienreihenfolge aufgedeckt werden, gibt es dafür Abhilfe. Sie setzen einfach hinter jeden `\item`-Befehl die Spezifikation $< +- >$. Das Pluszeichen steht für den internen Item-Zähler. Es geht aber noch einfacher, indem man die Spezifikation für die gesamte Aufzählung festlegt:

```
\begin{itemize}[<+->]
  \item Erster Punkt, auf jeder Folie.
  \item Ab der zweiten Folie.
  ...
\end{itemize}
```

Der Befehl `\alert` erlaubt die Hervorhebung einzelner Folienelemente (normalerweise durch rote Textfarbe). Indem man diesen Befehl mit einer numerischen Angabe kombiniert, lassen sich Elemente an bestimmten Stellen der Präsentation hervorheben:

```
\begin{itemize}
  \item<1-> \alert<1> {Erster Punkt, auf jeder Folie.}
  \item<2-> \alert<2> {Ab der zweiten Folie.}
  \item<3-> \alert<3> {Ab der dritten Folie.}
  ...
\end{itemize}
```

Mithilfe des `\action`-Befehls lassen sich mehrere Kommandos und Seitenangaben verbinden. Auf diese Weise lassen sich z. B. Elemente alternieren und zugleich hervorheben:

```
\begin{itemize}
  \item<+-| alert@+> Erster Punkt, auf jeder Folie.
  \item<+-| alert@+> Ab der zweiten Folie.
  \item<+-| alert@+> Ab der dritten Folie.
  ...
\end{itemize}
```

Beachten Sie das Leerzeichen hinter dem senkrechten Strich. Bei `alert@` können natürlich auch wieder alle möglichen Overlay-Spezifikationen hinter dem Klammeraffen stehen. Anstelle von *alert* sind auch noch folgende Aktionen möglich:

uncover: Anzeige des Items oder Blocks (Standardaktion)
only: Anzeige nur auf den spezifizierten Folien
visible: Text wird nur bei den spezifizierten Folien gezeigt.
invisible: Text wird nicht bei den spezifizierten Folien gezeigt.

Die Overlay-Spezifikationen können nicht nur bei Aufzählungen stehen, Sie können beispielsweise die folgenden Dinge realisieren:

```
\begin{itemize}
  % Fettdruck oder andere Hervorhebungen
  \textbf<2-4>{Dieser Text erscheint auf den Folien zwei,
               drei und vier fettgedruckt.}
  % Farbe
  \color<3->{green}{Dieser Text grün ab der dritten Folie.}
  % oder einfach nur eine LaTeX-Gruppe
  \begin{block}{Dieser Text ist ab Folie zwei zu sehen}<2->\end{block}

  ...
\end{itemize}
```

Beachten Sie aber, dass die Aufzählungsbefehle (`itemize` bzw. `enumerate`) beim Beamer-Paket nicht mehr identisch mit den Originalbefehlen sind.

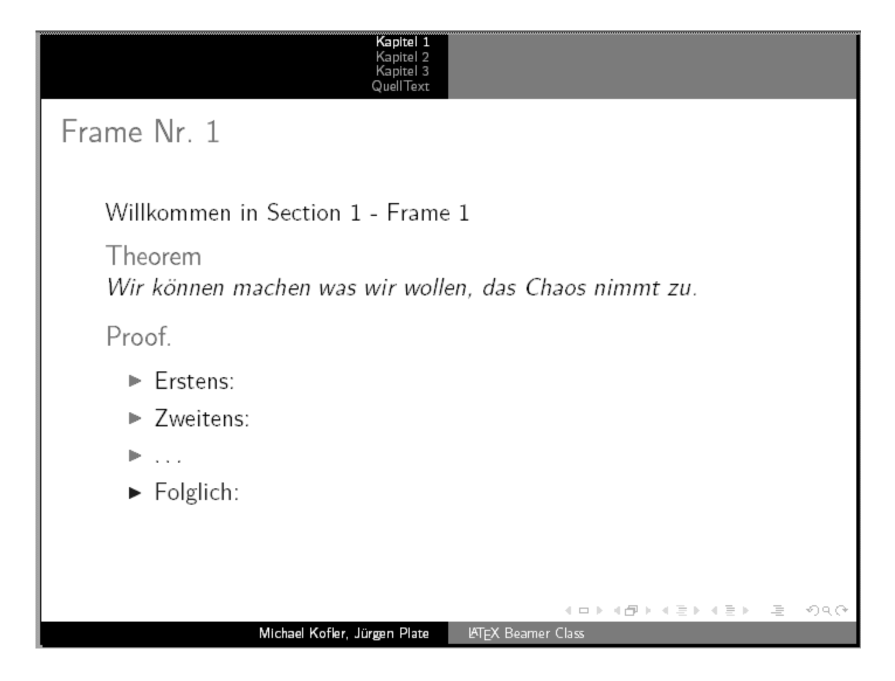


Abbildung 8.7: Eine Folie, erstellt mit dem Beamer-Paket

Hyperlinks

Im Beamer-Paket besteht – analog zu HTML – die Möglichkeit, explizite Links zu generieren. Hierzu werden Stellen im Dokument, auf die Sie referenzieren möchte, mit einer „Verankerung“ versehen. Dies geschieht folgendermaßen:

```
\hypertarget{Zielname}{beschreibender Text}
```

Auf die so erzeugten Verankerungen kann man an jeder Stelle des Dokumentes mittels

```
\hyperlink{Zielname}{beschreibender Text}
```

verweisen. Vollautomatisch werden von `pdflatex` intern Hypertargets für Einträge im Inhaltsverzeichnis gesetzt. Die Namen dieser Ziele kann man der beim LaTeX-Durchlauf entstandenen `.out`-Datei entnehmen. Im Beamer-Paket sind für die Navigation eine ganze Reihe von Hyperlink-Befehlen definiert. Bei den meisten Themen werden diese automatisch eingebunden und bieten diverse Navigationsmöglichkeiten per Mausklick.

Anstelle des textorientierten Links bietet das Beamer-Paket auch ein paar vordefinierte Buttons:

`\beamerbutton{Buttontext`: Ovaler, farbig unterlegter Button
`\beamergotobutton{Buttontext`: Dito mit Rechtspfeil
`\beamerskipbutton{Buttontext`: Dito mit doppeltem Rechtspfeil
`\beamerreturnbutton{Buttontext`: Dito mit Linkspfeil
`\beamerbutton{Buttontext`: Ovaler, farbig unterlegter Button
`\beamerbutton{Buttontext`: Ovaler, farbig unterlegter Button

Die Buttons werden anstelle des beschreibenden Textes bei `\hyperlink` eingesetzt.

Auf die gleiche Art und Weise kann auf externe Dokumente oder Adressen im Web verwiesen werden. Zu diesem Zweck gibt es die Befehle `\href` und `\url`. Beispiel:

```
Auf der \href{http://www.netzmafia.de/}{Website} finden Sie ...
```

Überblendeffekte

Das Beamer-Paket unterstützt die Überblendeffekte, die der Acrobat Reader beherrscht – es werden also nur die passenden PDF-Konstrukte durchgereicht. Deshalb ist mit solchen Effekten Vorsicht geboten, denn eventuell kann nicht jeder PDF-Betrachter alle darstellen. Außerdem kostet das Überblenden Zeit und wird auch schnell etwas nervig. Bei fast allen Effekten können Dauer und Richtung des Effekts als Optionen der Form `duration=sekunden` bzw. `direction=winkel` angegeben werden, wobei für den Winkel nur die Werte 0, 90,

180, 270 und für den „glitter“-Effekt 315 zulässig sind. Unter anderem gibt es folgende Effekte:

PDF-Überblendeffekte
`transdissolve`
`transboxout` und `transboxin`
`transblindshorizontal` und `transblindshorizontal`
`transsplithorizontalin` und `transsplithorizontalout`
`transsplitverticalin` und `transsplitverticalout`
`transglitter`
`transwipe`

Handouts etc.

Leider reicht der Platz hier nicht aus, die zahllosen Möglichkeiten des Beamer-Pakets umfassend zu beschreiben (immerhin ist die Originaldokumentation über 100 Seiten lang). Deshalb abschließend noch ein Tipp, wie Sie eine Papierversion für die Zuhörer erstellen können. Durch die Zeile

```
\documentclass[handout, ...]{beamer}
```

werden jeweils die Endzustände der interaktiven Folien dargestellt und man kann einen Ausdruck erzeugen. Beachten Sie das Seitenformat von nur 128 mm x 196 mm; beim Drucken ist da ein Vergößerungsfaktor zu wählen.

Genauso einfach ist die Produktion von Overhead-Folien (falls mal kein Beamer zur Verfügung steht). Da wird dann anstelle von „handout“ ganz einfach „trans“ eingesetzt. Übrigens sollte man für alle Fälle bei wichtigen Vorträgen auch einen Satz Folien mitnehmen – falls die Technik doch mal versagt.

8.11 LaTeX-Makros schreiben

Um das Erstellen eigener LaTeX-Befehle (Makros) kommen Sie früher oder später nicht herum, denn zum Einen können Sie kleine Makros als Abkürzung für lange Befehle schreiben, zum Anderen können Sie eine umfangreiche Eingabe mehrerer Befehle auf einen einzigen Befehl reduzieren. Dabei können Sie dann sogar Parameter übergeben und es gibt sogar optionale Argumente. Fangen wir aber mal ganz einfach an.

Definition neuer Kommandos

Die einfachste Art, ein Makro zu definieren, ist die folgende:

```
\newcommand{\befehl}{Makroinhalt}
```

Der neu zu definierende Befehl darf dabei noch nicht existieren. Gibt es den Befehl schon, kann er mit dem Befehl `\renewcommand` umdefiniert werden, der im Übrigen genauso arbeitet wie `\newcommand`.

Die Angabe des Makro-Inhaltes kann sowohl aus Text als auch aus beliebigen LaTeX-Befehlen bestehen. Ein Beispiel:

```
\newcommand{\ddk}{\textbf{Donauampfschiffahrtsgesellschaftskapitän}}
...
Auf der Kommandobrücke traf er auf den \ddk.
```

Schön wäre es, wenn der Text, der von dem Makro ausgegeben wird, variabel gestaltet werden könnte. Genau dazu können Sie die Makro-Parameter verwenden. Dazu wird der Befehl `\newcommand` um einen optionalen Parameter erweitert, der angibt, wie viele Parameter das Makro selbst haben soll:

```
\newcommand{\befehl}[Parameteranzahl]{Makroinhalt}
```

Sie müssen beim Aufruf Ihres Makros diese Anzahl von Parametern auch unbedingt mit angeben. Innerhalb des Makro-Inhaltes können die Parameter mithilfe von #1 #2 usw. angesprochen werden. Insgesamt können bis zu neun Parameter benutzt werden. Dazu ein Beispiel:

```
\newcommand{\bfit}[1]{\textbf{\textit{#1}}}
```

sorgt dafür, dass der übergebene Parameter fett und kursiv gedruckt wird. Sie können beispielsweise das Makro `\bfit{folgendermaßen}` benutzen.

Manchmal sind auch recht profane Makros wie die folgenden ganz nützlich, mit denen im Buch Kommandos bzw Dateiangaben gesetzt werden:

```
\newcommand{\komm}[1]{{\tt #1}}  % Schriftart für Unix-Kommandos
\newcommand{\dat}[1]{{\tt #1}}   % Datei- und Verzeichnisnamen
```

Natürlich könnte man das auch mittels `{\tt ...}` erledigen, aber wenn man nun auf die Idee käme, statt der Schreibmaschinenschrift Kursivschrift oder Kapitälchen zu verwenden, müsste man sich durch das gesamte Manuskript arbeiten, denn globales Suchen/Ersetzen würde ja auch andere Textteile betreffen, die in Schreibmaschinenschrift bleiben sollen. Die Verwendung von Makros macht solche Änderungen sehr leicht. Eine weite Anwendung liegt darin, bei der Rohfassung zunächst nur ein sehr einfaches Makro zu definieren, was dann später aufgemotzt werden kann.

Wenn Sie mit eigenen Makros experimentieren, kann es vorkommen, dass sich bei deren Anwendung unerwünschte Leerzeichen zeigen. Das rührt meist von Leerzeichen oder Zeilenumbrüchen bei der Definition des Makros her. Schreiben Sie daher Ihre Makros so kompakt wie möglich. Die Zeilenwechsel lassen sich durch ein %-Zeichen am Ende der Zeile „entschärfen“.

Im folgenden einfachen Beispiel sollen verschieden große quadratische Kästchen definiert werden. Die Namensgebung lehnt sich an die der Schriftgrößen an:

```
\newcommand{\bull}{\rule{0.8ex}{0.8ex}}
\newcommand{\Bull}{\rule{1ex}{1ex}}
\newcommand{\BULL}{\rule{1em}{1em}}
```

Damit kann man nun die folgenden Kästchen ausgeben:
`\bull` ■ `\Bull` ■ `\BULL` ■
und in `\Large`: `\bull` ■ `\Bull` ■ `\BULL` ■ .

Probieren wir noch ein Beispiel: Es soll ein Formular gestaltet werden, das bei Eingabefeldern Kästchen für jeden Buchstaben bzw. jede Ziffer hat. Da können wir das verwenden, was auf Seite 367 bei den Boxen erwähnt wurde:

```
\newsavebox{\KKBox}% Neue Box benennen und definieren
% \rule legt die Höhe, \hspace die Breite des Kästchens fest
\sbox{\KKBox}{\fbox{\rule{0mm}{1em}\hspace{1ex}}}
\newcommand{\KK}{\usebox{\KKBox}}

% dann eine Eingabeanforderung:
Bankleitzahl: \KK\KK\KK\quad\KK\KK\KK\quad\KK\KK \\
Kontonummer: \KK\KK\KK\quad\KK\KK\KK\quad\KK\KK\KK\quad\KK\KK\KK \\
```

Das Ergebnis stellt sich dann folgendermaßen dar:

Bankleitzahl: ☐☐☐ ☐☐☐ ☐☐
Kontonummer: ☐☐☐ ☐☐☐ ☐☐☐ ☐☐☐

Möchten Sie ein eigenes Makro definieren, bei dessen Aufruf der erste Parameter optional ist, also in eckigen Klammern angegeben wird, gehen Sie folgendermaßen vor:

```
\newcommand{\befehl}[Parameterzahl][Default]{Makroinhalt}
```

Bei der Parameterzahl wird der optionale Parameter natürlich mitgezählt. Wird beim Aufruf des Makros der optionale Parameter angegeben, so wird er Platzhalter #1 zugewiesen, alle anderen Parameter den Platzhaltern #2, #3 usw. Im anderen Fall nimmt #1 den Default-Wert an. Auch dazu ein Beispiel:

```
\newcommand{\Name}[3][Herr]{#1 #2 {\bf #3}}

...
\Name{Donald}{Duck},
\Name{Dagobert}{Duck},
\Name[Frau]{Daisy}{Duck}
...
```

Das ergibt:

...Herr Donald **Duck**, Herr Dagobert **Duck**, Frau Daisy **Duck** ...

Problematisch kann auch die Verwendung des Mathematik-Modus sein, denn das Makro kann ja im normalen Absatz-Modus genauso gut aufgerufen werden, wie in der Mathematik-Umgebung. Abhilfe bietet hier das Makro `\ensuremath`. In der normalen Umgebung sorgt es dafür, das in den Mathe-Modus umgeschaltet wird (und nacher wieder zurück), ansonsten macht es gar nichts. Der neue Befehl `\PYT` im folgenden Beispiel kann sowohl im Mathe-Modus als auch im Absatz-Modus verwendet werden:

```
\newcommand{\PYT}{\ensuremath{a^2+b^2=c^2}}
...
\[ \PYT \]
...
Pythagoras hat herausgefunden: \PYT
```

liefert in beiden Fällen die richtige Formelausgabe:

...

$$a^2 + b^2 = c^2$$

...Pythagoras hat herausgefunden: $a^2 + b^2 = c^2$

Die Definition eigener Umgebungen

Als Umgebung oder Environment bezeichnet man in LaTeX Befehlsumgebungen, die mit `\begin{...}` und `\end{...}` geklammert werden. Mit dem Befehl

```
\newenvironment{name}[parameteranzahl]{begin-befehle}{end-befehle}
```

können Sie sich solche Umgebungen einschließlich Parameterübergabe selbst definieren. Für die Parameter gelten die gleichen Regeln wie bei der Neudefinition von Kommandos.

Legen Sie einen neuen Namen für Ihre Umgebung fest und definieren Sie anschließend die Befehle, die jeweils beim Eintritt (`\begin{...}`) und Austritt (`\end{...}`) der Umgebung ausgeführt werden sollen. Zu übergebende Parameter müssen beim Eintreten in die Umgebung angegeben werden. Ihre Umgebung können Sie dann mittels `\begin{Name}{Parameter}` ... `\end{Name}` benutzen. Ein Beispiel:

```
% Umgebung fuer Listings und andere Textteile, die 'verbatim'
% ausgegeben werden sollen. Schrift kleiner (footnotesize),
% TT-Font, Abstand zum Text etwas kleiner als bei Standard-verbatim.
\newenvironment{listing}%
 {\vspace{-6pt}\footnotesize\verbatim}% begin-Befehle
 {\endverbatim\vspace{-6pt}}% end-Befehle
```

Das Beispiel macht sich auch zunutze, dass bei der `verbatim`-Umgebung die Makros `\verbatim` (für `\begin{verbatim}`) und `\endverbatim` (für `\end{verbatim}`) zur Verfügung stehen.

Ein weiteres Beispiel soll das Umfeld einer Tabelle vereinfachen. Der neuen Umgebung `tabelle` werden drei Parameter mitgegeben, die Tabellenspalten-Definition, die Tabellenüberschrift (caption) und ein Label, damit man im Text auf die Tabelle verweisen kann:

```
\newenvironment{tabelle}[3]%
 {\begin{table}[h!t]
  \renewcommand{\arraystretch}{1.20}% Abstände etwas groesser
  \begin{center}
  \caption{#2 \label{#3}}
  \vspace{3pt}
  \begin{tabular}{#1}}% ende begin-Befehle
 {\end{tabular}
  \end{center}
  \end{table}}%
```

Umdefinition von Befehlen und Umgebungen

Auch hier gelten die Regeln wie bei der Neudefinition von Makros. Mit Vorsicht ist jedoch an das Umdefinieren von LaTeX-Befehlen und -Umgebungen heranzugehen. Eine Umdefinition eigener Kommandos ist natürlich problemlos. Die Kommandos zur Umdefinition lauten `\renewcommand` und `\renewenvironment`.

Sie haben den gleichen syntaktischen Aufbau und unterliegen den gleichen Regeln wie die Kommandos zur Neudefinition von Makros und Umgebungen. Jedoch muss als Name ein bereits existierendes Kommando eingegeben werden. Sinnvoll und erwünscht sind zum Beispiel die Umdefinitionen von `\baselinestretch`, `\arraystretch` und anderen Größenangaben, zum Beispiel:

```
% Darstellung der Bilder im Text aendern
% Bildunterschriften einen Punkt kleiner setzen
\renewcommand{\captionfont}{\small}
\renewcommand{\captionlabelfont}{\small \bf}
% Bildunterschriften mit 'Bild' statt 'Abbildung'
\renewcommand{\figurename}{Bild }
```

Der \newtheorem-Befehl

`\newtheorem` erzeugt eine Umgebung für Lehrsätze (Theoreme), Definitonen usw. Bei diesem Befehl gibt es gleich drei Syntaxvarianten:

```
\newtheorem{Name}{Überschrift}
\newtheorem{Name}{Überschrift}[KapZähler]
\newtheorem{Name}[ThmZähler]{Überschrift}
```

Der Aufruf im Text erfolgt wie bei anderen Umgebungen: `\begin{Name} ... \end{Name}`. Es wird eine abgesetzte Umgebung mit Überschrift und einem Zähler erzeugt – wie beispielsweise bei Kapiteln und Abschnitten. Das Verhalten des Zählers hängt von den optionalen Parametern *KapZähler* bzw. *ThmZähler* ab, deren Wirkung sich unterscheidet:

- Ohne Verwendung der optionalen Zähler, werden alle Vorkommen der `theorem`-Umgebungen der Reihe nach durchnummeriert (Satz 1, Satz 2 usw.).
- Bei Angabe des Parameters *KapZähler* (z. B. chapter, section usw.) wird die Nummer der Umgebung zurückgesetzt, wenn der entsprechende Gliederungsbefehl aufgerufen wird. Man kann die Theoreme also kapitel- oder abschnittsweise durchzählen.
- Bei Angabe des Parameters *ThmZähler* (der Name einer vorher definierten anderen `theorem`-Umgebung) erhält die neue Umgebung keine eigene Nummerierung, sondern wird mit der angegebenen Umgebung mitgezählt.

Beispiel:

```
\newtheorem{these}{These}
\newtheorem{folg}{Folgerung}[chapter]

\begin{these}[Plate]
Wir können machen, was wir wollen, die Unordnung nimmt zu.
(Abgeleitet aus dem 2. Gesetz der Thermodynamik)
\end{These}

\begin{folg}
Aufräumen führt nur zu Unordnung an einer anderen Stelle
im Universum.
\end{folg}
```

Das ergibt:

These 1 (Plate) *Wir können machen, was wir wollen, die Unordnung nimmt zu. (Abgeleitet aus dem 2. Gesetz der Thermodynamik)*

Folgerung 1 *Aufräumen führt nur zu Unordnung an einer anderen Stelle im Universum.*

LaTeX bietet eine Fülle weiterer Möglichkeiten, unter anderem das Erstellen von Zeichnungen oder exzessive Möglichkeiten der Textgestaltung. Wir wollen

jedoch an dieser Stelle mit der Einführung in die Arbeit mit LaTeX schließen, denn erstens kann das Buch nicht beliebig dick werden und zweitens bietet das bisher Gesagte mehr als genug Grundlagen für alle studentischen Belange.

Erlauben Sie uns noch ein paar Schlussbemerkungen, bevor wir auf die Tools zur Bearbeitung des Quelltextes eingehen:

- Texte ohne WYSIWYG zu erstellen, erfordert ein gewisses Umdenken. Sie werden aber nach einiger Zeit merken, dass die Gedanken viel schneller und klarer „zu Papier" gelangen, wenn man sich erst mal nicht um das Aussehen kümmern muss.
- Halten Sie durch! Zu Beginn ist die Frustration, die durch die scheinbar unverständlichen Fehlermeldungen erzeugt wird, relativ hoch. Mit der Zeit werden die Fehler weniger und Sie verstehen auch, was LaTeX Ihnen sagen will.
- Aus den Quelltexten können Sie auch in zehn Jahren noch ein Buch oder einen Artikel weiterbearbeiten und in hoher Qualität ausdrucken – versuchen Sie das mal mit den Dateien, die mit einer beliebigen Textverarbeitung erstellt wurden.
- Schlussendlich bietet Ihen LaTeX die Möglichkeit, den Quelltext per Programm zu erzeugen. Sie können also per Programm absolut „schicke" Dokumente generieren.

8.12 LaTeX-Dokumente anzeigen und weiterverarbeiten

8.12.1 DVI-Dateien anzeigen (xdvi, kdvi)

Das Kommando `latex` erzeugt aus Ihrer LaTeX-Datei eine DVI-Datei. Mit den Programmen `xdvi` und `kdvi` können Sie `*.dvi`-Dateien auf dem Bildschirm ansehen und darin blättern. Denken Sie daran, dass einige PostScript-spezifische Gestaltungsmöglichkeiten in `xdvi` und `kdvi` nicht angezeigt werden können. Um ein LaTeX-Dokument exakt so zu sehen, wie es ausgedruckt wird, müssen Sie es in das PostScript-Format umwandeln und mit `ghostview`, `gv`, `ggv` oder `kghostview` betrachten.

Wenn das LaTeX-Dokument die LaTeX-eigenen Schriften verwendet, müssen deren Bitmaps bei der ersten Verwendung erzeugt werden. Dazu starten `xdvi` bzw. `kdvi` automatisch das LaTeX-Zusatzprogramm `metafont`. Deswegen kann das erstmalige Anzeigen einer DVI-Datei relativ lange dauern.

`xdvi` wird mit dem Dateinamen der `*.dvi`-Datei als Parameter gestartet. Anschließend kann mit den Buttons *Next*, *Previous* etc. in dem Dokument geblättert werden. Wenn Sie die Maus über den Bildausschnitt bewegen und dabei

die linke Taste drücken, wird (verzögerungsfrei) ein vergrößerter Ausschnitt des Rechtecks unter der Maus dargestellt. Die folgende Tabelle enthält eine Übersicht der wichtigsten Tastenkürzel zur Steuerung von `xdvi`.

xdvi-Tastenkürzel	
Bild ↑, Bild ↓	vorige/nächste Seite
Leertaste, ↵	nächste Seite
Backspace, Entf	vorige Seite
↑, ↓, ←, →	Bildausschnitt verschieben
G	springt zur vorher eingegebenen Seitennummer
V	(de)aktiviert die Anzeige von PostScript-Grafiken

`kdvi` muss bei manchen Distributionen extra installiert werden. Bemerkenswert ist die komfortable Export-Funktion für die Formate PostScript und PDF, die den manuellen Aufruf von `dvips` bzw. `dvipdfm` erspart.

8.12.2 PostScript-Dokumente erzeugen (dvips)

`dvips` wandelt `*.dvi`-Dateien in das PostScript-Format um. Die Syntax des Kommandos sieht so aus:

`user$` **dvips [optionen] -o name.ps name.dvi**

-A wandelt nur ungerade Seiten um.

-B wandelt nur gerade Seiten um.

-D *n* verwendet bei der Erzeugung von LaTeX-Bitmap-Schriften eine Auflösung von *n* dpi (*dots per inch*). Die Standardauflösung beträgt meist 600 dpi. Alternativ darf *n* auch 300, 400 oder 1270 (für die Druckerei) betragen. Die Option ist nur für Bitmap-Schriften relevant. Eingesetzte PostScript-Schriften sind immer auflösungsunabhängig.

-E erzeugt eine EPS-Datei (*Encapsulated PostScript*) mit einer BoundingBox, die nur den tatsächlich genutzten Teil der Seite umfasst. Das ist nur sinnvoll, wenn die DVI-Datei nur eine Seite hat und die resultierende EPS-Datei anschließend in ein anderes Dokument eingebettet werden soll.

-G0 verhindert Inkompatibilitäten mit dem Adobe Reader. Diese Option ist nur zweckmäßig, wenn die PostScript-Datei später in eine PDF-Datei umgewandelt werden soll.

-i -S *n* zerlegt die Ausgabe in Dateien zu je *n* Seiten. Die Dateien werden automatisch durchnummeriert.

-l *letzteseite* beendet die Umwandlung mit der angegebenen Seite.

-o *zieldatei* schreibt das Ergebnis in die angegebene Datei (anstatt es an das Programm `lpr` weiterzuleiten).

-p *ersteseite* beginnt die Umwandlung mit der angegebenen Seite.

-pp *n1,n2-n3,n4,n5,n6-n7* druckt die angegebenen Seiten. Beachten Sie, dass in der Seitenliste keine Leerzeichen vorkommen dürfen.

-P*name* berücksichtigt zusätzliche `dvips`-Default-Einstellungen. Beispielsweise bewirkt `-Ppdf`, dass die resultierende Datei für eine spätere Umwandlung in ein PDF-Dokument optimiert wird (Konfigurationsdatei `/var/lib/texmf/dvips/config/config.pdf`).

Das Kommando kann auch einfach als `dvips name` ausgeführt werden. Es liest dann `name.dvi` und schreibt das Ergebnis in `name.ps`. Globale Default-Einstellungen für `dvips` sind in `/etc/texmf/config.ps` bzw. in `/usr/share/texmf/dvips/config.ps` definiert.

Probleme mit pixeligen Schriften: In der Vergangenheit sahen aus LaTeX-Dokumenten erzeugte PostScript- und PDF-Dokumente oft pixelig aus. Der Grund besteht darin, dass die LaTeX-Originalschriften tatsächlich Bitmap-Schriften sind. Bei aktuellen teTeX-Versionen sollte dieses Problem nicht mehr auftreten, weil es nun auch zu den LaTeX-Originalschriften PostScript-Varianten gibt, die per Default eingesetzt werden. Verantwortlich dafür ist die Datei `/usr/share/texmf/dvips/tetex/bsr.map`, die von `dvips` automatisch berücksichtigt wird.

Wenn Sie bei Ihrer LaTeX-Version dennoch mit diesem Problem kämpfen, gibt es folgende Lösungswege:

- Stellen Sie das gesamte Dokument auf PostScript-Schriften um (siehe Seite 429). Dazu reichen im Regelfall zwei oder drei `\usepackage`-Zeilen. Allerdings ändert sich durch diese Maßnahme der Zeilen- und Seitenumbruch.
- Falls Sie bei den LaTeX-Standardschriften bleiben möchten, entfernen Sie die Anweisung `\usepackage[T1]{fontenc}`. Momentan stehen PostScript-Varianten für die LaTeX-Standardschriften nur in der LaTeX-Font-Kodierung (CM-Schriften) zur Verfügung, nicht aber für die T1- und TS1-Kodierung (EC- und TC-Schriften). Hintergrundinformationen zu den CM-, EC- und TC-Schriften sowie zur Font-Codierung finden Sie auf Seite 428.
- Fügen Sie die folgende Zeile in `config.ps` ein:

  ```
  p +bsr.map
  ```
- Erhöhen Sie die Auflösung der Bitmap-Schriften mit der `dvips`-Option `-D`. Die Druckqualität steigt dadurch deutlich an. (Die Darstellung im Adobe Reader bleibt leider dennoch spürbar pixelig; es ist aber ein erstklassiger Ausdruck möglich.)

8.12.3 PDF-Dokumente erzeugen

Oft möchte man LaTeX-Dokumente als PDF-Datei weitergeben. Dazu gibt es eine ganze Menge Möglichkeiten. In allen Fällen erhalten Sie als Ergebnis eine PDF-Datei, die wie die äquivalente PostScript-Datei aussieht. Ob auch PDF-Zusatzfunktionen (Inhaltsverzeichnis, anklickbare Links etc.) genutzt werden können, hängt vom beschrittenen Umwandlungsweg und von den im LaTeX-Dokument eingesetzten Zusatzpaketen ab.

- Sie erzeugen zuerst mit `dvips` eine PostScript-Datei und wandeln diese dann mit `ps2pdf` oder mit dem Adobe Distiller in eine PDF-Datei um. (Adobe Distiller ist Teil des kommerziellen Programmpakets Adobe Acrobat, von dem es zurzeit leider keine Linux-Version gibt.) PDF-Funktionen können durch das LaTeX-Paket *hyperref* genutzt werden.
- Sie wandeln die DVI-Datei mit `dvipdfm` in eine PDF-Datei um. Für PDF-Funktionen müssen Sie zusätzliche `\special`-Kommandos in das LaTeX-Dokument einfügen und *hyperref* nutzen.
- Sie wandeln die LaTeX-Datei mit `pdflatex` direkt in eine PDF-Datei um (wie schon im Abschnitt 8.10 beschrieben). Dieses Programm sieht eine Reihe zusätzlicher LaTeX-Kommandos vor, um die PDF-Funktionen zu steuern.

dvipdfm fasst einige Teilschritte in einem Programm zusammen und kann einfach als `dvipdfm name` ausgeführt werden. Es liest dann `name.dvi` und schreibt das Ergebnis in `name.pdf`. Das Programm `dvipdfm` ist vor allem dann interessant, wenn die in der Dokumentation (Datei `dvipdfm.pdf`) beschriebenen `\special`-Kommandos eingesetzt werden:

`user$` **dvipdfm [options] name.dvi $¿$ name.pdf**

-l verwendet das Querformat (*landscape*).

-p *papersize* verwendet das angegebene Papierformat (z. B. letter, legal, a3, a4 oder a5).

-r *dpi* verwendet den angegebenen dpi-Wert bei der Erzeugung von Bitmap-Fonts der LaTeX-Schriften.

-s *pages* gibt die gewünschten Seiten an (z. B. `1,3,7,9-12`).

-z *n* gibt den gewünschten Kompressionsgrad (0-9) an. 9 bedeutet maximale Kompression.

pdflatex ist eine Variante zu LaTeX, die speziell dafür entwickelt wurde, PDF-Dateien zu erzeugen. Dementsprechend gibt es eine Reihe zusätzlicher Kommandos. Die größte Einschränkung besteht darin, dass `pdflatex` nicht mit allen LaTeX-Erweiterungen zurechtkommt. Das folgende Kommando liefert als Ergebnis direkt die PDF-Datei `name.pdf`:

```
user$ pdflatex name.tex
```

Weitere Informationen und Beispiele finden Sie in den Dateien des Verzeichnisses `/usr/share/texmf/doc/pdftex/base`. Die zentrale Anlaufstelle im Internet finden Sie unter *http://www.tug.org/applications/pdftex/*.

hyperref: Das `hyperref`-Paket hilft, die Möglichkeiten des PDF-Formats besser zu nutzen. Die folgenden Zeilen am Beginn eines LaTeX-Dokuments bewirken, dass das PDF-Dokument mit einem PDF-kompatiblen Inhaltsverzeichnis ausgestattet wird (ausklappbare Bookmarks) und alle Querverweise innerhalb des Dokuments blau hervorgehoben werden. Die Querverweise können per Maus angeklickt werden.

```
\usepackage[ps2pdf]{hyperref}
\hypersetup{colorlinks=true, linkcolor=darkblue, urlcolor=blue}
```

Querverweise in das Internet können mit `\url{http://www.adresse.com}` oder mit `\href{http://adresse}{beschreibender Text}` formuliert werden und sind dann ebenfalls anklickbar.

Das `hyperref`-Paket ist mit `dvipdfm` inkompatibel. Die Funktionen des `hyperref`-Pakets sind wirksam, wenn Sie die PDF-Datei mit `pdflatex` `dvipdf` oder mit `dvips` und `ps2pdf` erzeugen!

`hyperef` bietet noch viel mehr Zusatzfunktionen, die unter anderem im Buch *Mit LaTeX ins Web* von Michael Goossens und Sebastian Rahtz dokumentiert sind. Einen exzellenten Überblick über die zahlreichen Möglichkeiten, LaTeX-Texte in das PDF-Format umzuwandeln, gibt das PDF-Dokument *http://www.ctan.org/tex-archive/info/german/LaTeX2PDF.pdf*.

8.12.4 HTML-Dokumente erzeugen

Wenn Sie HTML-Dateien aus LaTeX-Dokumenten erstellen möchten, bieten sich zwei Werkzeuge an: `latex2html` und `tex4ht`.

Bei **latex2html** handelt es sich um ein Perl-Script, das aus einer LaTeX-Datei eine oder mehrere `*.html`-Dateien erzeugt. Formeln und LaTeX-Sonderzeichen werden in `*.gif`-Bilder übersetzt. Dazu werden das Grafikpaket `netpbm` sowie `gs` eingesetzt.

`latex2html` zerlegt das Dokument in einzelne Abschnitte und speichert alle resultierenden HTML- und PNG-Dateien im neuen Verzeichnis `name`. Das Programm funktioniert nur dann wunschgemäß, wenn nur LaTeX-Standardkommandos eingesetzt werden (nicht aber diverse eigene Makros, Zusatzpakete etc.). Meist funktioniert die Übersetzung nur bei einfacheren Texten. Weitere Informationen zu `latex2html` finden Sie unter *http://www.latex2html.org/*.

tex4ht muss in das LaTeX-Dokument mittels `\usepackage{tex4ht}` eingefügt werden. Anschließend wird der Text mit dem Kommando `ht latex [options] name.tex` übersetzt. Als Resultat erhalten Sie die Datei `name.html` und

eventuell eine Reihe weiterer HTML-Dateien. Als Basis für die Umwandlung in das HTML-Format dient eine DVI-Datei. `tex4ht` ist ausführlich im oben erwähnten Buch *Mit LaTeX ins Web* beschrieben. Einen Überblick über die Konfigurationsmöglichkeiten gibt die Webseite *http://www.cis.ohio-state.edu/ xgurari/TeX4ht/*.

8.13 Metafont- und PostScript-Schriften

Der Umgang mit Schriftarten bereitet bei der Arbeit mit LaTeX vermutlich die größten Verständnisprobleme. Dieser Abschnitt versucht, ein wenig Klarheit zu schaffen. Die zwei zentralen Themen dieses Abschnitts sind das LaTeX-Zusatzprogramm `metafont`, mit dem die LaTeX-eigenen Schriften erzeugt werden, und die Verwendung von PostScript-Schriften an Stelle der LaTeX-Originalschriften. Noch viel mehr Informationen über die Hintergründe von MetaFont- und PostScript-Schriften finden Sie in den deutschen TeX-FAQs unter *http://www.dante.de/faq/de-tex-faq/*.

8.13.1 Metafont-Schriften

Als LaTeX ursprünglich entwickelt wurde, war der Druckermarkt noch unübersichtlicher als jetzt. Einen etablierten Standard wie PostScript mit zahllosen vordefinierten Schriftarten zu einem für (beinahe) jedermann erschwinglichen Preis gab es damals noch nicht. Aus diesem Grund wurde gleichzeitig mit LaTeX das Programm `metafont` entwickelt. Dieses Programm ist für die Berechnung der Schriften zuständig. Das Ziel war es, möglichst jeden beliebigen Drucker in optimaler Qualität unterstützen zu können. Vor diesem Hintergrund ist auch zu verstehen, warum `metafont` mit Bitmap-Schriften (und nicht wie PostScript mit beliebig skalierbaren Vektorschriften) arbeitet. LaTeX und das Metafont-System bilden bis heute ein integratives Paket; jedes Programm für sich ist praktisch wertlos.

Wie funktioniert Metafont? Ausgangspunkt für alle Schriften sind `name.mf`-Dateien. Diese Textdateien enthalten Kommandos zum Zeichnen der einzelnen Buchstaben einer Schriftart. Das Programm `metafont` erzeugt je nach den angegebenen Optionen eine oder zwei Dateien: in jedem Fall eine Bitmap-Datei `name.nnnpk` und manchmal (wenn diese Datei noch nicht existiert) die Metrikdatei `name.tfm`.

Die Bitmap-Dateien (`*.nnnpk`) enthalten die Schrift in komprimierter Form. In dieser Bitmap ist jeder Buchstabe durch Tausende von Einzelpunkten (Pixeln) dargestellt. `nnn` steht dabei für einen Faktor aus Auflösung in dpi (dots per inch) und Vergrößerung. Typische dpi-Werte sind 300 oder 600 dpi bei Laserdruckern sowie 1270 oder 2540 dpi bei Belichtungsgeräten. Der Vergrößerungsfaktor kommt ins Spiel, wenn eine Schrift in einer anderen Größe als in ihrer

Entwurfsgröße benötigt wird (z. B. bei `\small` oder `\large`). Daraus ergeben sich dann Werte wie 720 (600 mal 1,2).

Die Metrikdateien (`*.tfm`) enthalten alle Angaben über die Größen der einzelnen Buchstaben. Auch wenn es zu einer Schrift mehrere `*pk`-Dateien gibt, existiert immer nur eine Metrikdatei (die wahre Größe der Buchstaben ist ja von der Druckerauflösung unabhängig).

Nun zur realen Bedeutung dieser beiden Dateitypen: Die Metrikdateien `*.tfm` werden während der Bearbeitung eines Textes durch LaTeX benötigt. LaTeX entnimmt diesen Dateien die Information, wie groß die jeweiligen Buchstaben sind, und führt anhand dieser Daten den Zeilen- und Seitenumbruch durch. Das Ergebnis ist eine DVI-Datei (*device independent*), die die eigentlichen Schriften nicht enthält und in dieser Form weder ausgedruckt noch angezeigt werden kann.

Die `*pk`-Dateien werden erst beim Ausdruck bzw. bei der Anzeige der Datei am Bildschirm (`xdvi`) benötigt. Wenn die gerade erforderliche `*pk`-Datei in der Auflösung des Druckers bzw. Bildschirms und in der gewünschten Vergrößerung noch nicht existiert, wird `metafont` automatisch gestartet. Deswegen kann es beim ersten Ausdruck eines Textes mit vielen Schriftarten und -größen zu erheblichen Verzögerungen kommen. Beim nächsten Mal stehen die erforderlichen `*pk`-Dateien dann aber bereits zur Verfügung.

Im Regelfall müssen Sie sich nicht um Schriftartdateien kümmern und brauchen das Programm `metafont` (Kommandoname `mf`) nie selbst aufzurufen. Alle erforderlichen `*.mf`- und `*.tfm`-Dateien sind bereits vorinstalliert und die `*.nnnpk`-Dateien werden je nach Bedarf automatisch erzeugt.

CM-Schriften: Unter LaTeX kommen per Default vier Familien der CM-Schriften zum Einsatz: CM Roman (Standardschrift), CM Sans Serif, CM Typewriter und eine CM-Roman-ähnliche Schrift für mathematische Zeichen. (CM steht für *computer modern*. Die CM-Schriften wurden vom LaTeX-Erfinder und -Entwickler Donald Knuth entworfen.)

Die CM-Schriften bestehen nur aus den 128 Zeichen des ASCII-Zeichensatzes (bzw. ISO-7-Bit). Buchstaben wie å, é, ñ oder ö sind keine eigenen Zeichen, sondern werden durch die Überlagerung unterschiedlicher Zeichen gebildet. Die Zuordnung zwischen Zeichen und Codes wird als OT1-Kodierung bezeichnet.

Da dies typografisch nicht optimal war, wurden die **EC-Schriften** geschaffen, in denen fast alle in europäischen Ländern üblichen Buchstaben als eigene Zeichen enthalten sind (insgesamt 256 Zeichen). Um diese Schriften zu nutzen, müssen Sie die Zeile `\usepackage[T1]{fontenc}` in Ihr LaTeX-Dokument einfügen.

Diese Anweisung verändert rein optisch fast nichts am Aussehen Ihrer mit LaTeX gesetzten Dokumente. Sie müssten schon ein Typografie-Experte sein, um Unterschiede zu bemerken. Allerdings sind nun viele Zeichen in den DVI-Dateien anders codiert (T1-Kodierung). Das gilt auch für daraus resultierende PDF-Dateien. Ein wesentlicher Vorteil der T1-Kodierung besteht darin, dass Sie nun in PDF-Dokumenten auch nach Wörtern mit den Buchstaben ä ö ü ß oder anderen Buchstaben außerhalb des ASCII-Zeichensatzes suchen können.

Manche häufig in Texten vorkommende Zeichen fanden allerdings auch in den EC-Schriften nicht Platz. Als Ergänzung zu den EC-Schriften wurden daher die **TC-Schriften** entworfen, die zusätzliche Zeichen enthalten. Für die Zuordnung zwischen Zeichen und Kodes gilt die TS1-Kodierung. Um auch diese Schriften nutzen zu können, fügen Sie die Zeile `\usepackage{textcomp}` in Ihr LaTeX-Dokument ein.

Es stehen Ihnen nun eine Reihe neuer `\textxxx`-Kommandos für diverse Sonderzeichen zur Verfügung, z. B. `\textcurrency` für ¤ oder `\textbrokenbar` für ¦.

8.13.2 PostScript-Schriften (Type-1-Fonts)

Die auf Metafont basierenden Bitmap-Schriften waren zwar 1985 (LaTeX 2.09) richtungsweisend, sie sind heute aber nicht mehr zeitgemäß. Zur optimalen Weiterverarbeitung von LaTeX-Texten ist es wünschenswert, die Bitmap-Schriften durch PostScript-Schriften (so genannte Type-1-Fonts) zu ersetzen. Das macht qualitativ optimale Ausdrucke möglich (etwa für den Buchdruck) und erleichtert die Weitergabe im PDF-Format ohne Pixelartefakte.

PostScript-Ersatz für die CM-Schriften: Die aktuelle teTeX-Version enthält für die CM-Schriften äquivalente PostScript-Schriften (Verzeichnis `/usr/share/texmf/fonts/type1/bluesky/`). Diese Schriften werden beim Aufruf von `dvips` automatisch eingesetzt. Insofern ist das Problem der Bitmap-Schriften eigentlich schon gelöst, ohne dass Sie irgendetwas tun müssen. Die PostScript-Schriften sind durch eine Umwandlung der originalen Metafont-Schriften entstanden und sollten absolut identisch aussehen.

Das einzige Problem an dieser scheinbar optimalen Lösung besteht darin, dass es momentan noch keinen PostScript-Ersatz für die EC- und TC-Schriften gibt. Deswegen funktioniert die automatische Ersetzung der LaTeX-Schriften durch PostScript-Schriften *nicht*, wenn sich die beiden folgenden Zeilen im LaTeX-Dokument befinden:

```
\usepackage[T1]{fontenc} % T1-Codierung statt OT1-Codierung
\usepackage{textcomp}    % Sonderzeichen mit TS1-Codierung
```

PostScript-Varianten für die EC- und TC-Schriften sind im Internet bereits verfügbar. Sie werden wahrscheinlich in die teTeX-Distribution aufgenommen, sobald sie vollständig ausgereift sind.

PostScript-Standardschriften einsetzen: Ein anderer Lösungsweg besteht darin, auf die CM-Schriften ganz zu verzichten und stattdessen auf PostScript-Standardschriften wie Times, Palatino, Helvetica oder Courier umzusteigen. Erfreulicherweise gibt es zu diesem Zweck fertige LaTeX-Pakete (Sammelbegriff PSNFSS2e). Sie müssen lediglich ein paar `\usepackage`-Anweisungen in Ihr LaTeX-Dokument einfügen. Beachten Sie aber, dass sich dadurch der Zeilen- und Seitenumbruch Ihres Dokuments ändern kann.

```
\usepackage{mathpazo}    % Palatino statt der LaTeX-Originalschrift
```

In LaTeX-Dokumenten gibt es immer vier Schriftfamilien (Standardschrift, Sans Serif, Typewriter, Mathematik-Schrift). Die folgenden Pakete ersetzen aber jeweils nur einzelne Schriften und lassen die anderen unverändert. Wenn Sie alle LaTeX-Schriften durch PostScript-Schriften ersetzen möchten, müssen Sie mehrere Pakete kombinieren (z. B. *mathptmx*, *helvet* und *courier*)!

PostScript-Font-Pakete	
`avant`	AvantGarde statt Sans Serif, andere Schriften: Original-LaTeX
`bookman`	Bookman als Standardschrift, AvantGarde statt Sans Serif, Courier statt Typewriter
`chancery`	Zapf Chancery als Standardschrift, andere Schriften: Original-LaTeX
`charter`	Charter als Standardschrift, andere Schriften: Original-LaTeX
`courier`	Courier statt Typewriter, andere Schriften: Original-LaTeX
`helvet`	Helvetica statt Sans Serif, andere Schriften: Original-LaTeX
`mathptmx`	Times als Standardschrift, Times-ähnliche Mathe-Schrift, andere Schriften: Original-LaTeX
`mathpazo`	Palatino als Standardschrift, Palatino-ähnliche Mathe-Schrift, andere Schriften: Original-LaTeX
`newcent`	Standardschrift New Century Schoolbook, AvantGarde statt Sans Serif, Courier statt Typewriter
`utopia`	Utopia als Standardschrift, andere Schriften: Original-LaTeX

Das Paket *mathptmx* unterstützt keine Formeln in fetter Schrift (d. h. `\boldmath` steht nicht zur Verfügung).

Die Schriften Utopia und Charter sind zwar frei verfügbar, zählen aber nicht zu den 35 PostScript-Standardschriften von Adobe. Es kann daher sein, dass diese Schriften extra installiert werden müssen.

Um das Ergebnis weiter zu verbessern, sollten Sie bei den meisten Schriften den Durchschuss (den Zeilenabstand) mit `\linespread{n}` optimieren. Dabei ist n ein Faktor, der den Zeilenabstand vergrößert oder verkleinert. Bei vielen Schriften verbessert ein Wert zwischen 1.05 und 1.1 die Lesbarkeit.

Weil die Schrift Helvetica etwas größer als die meisten anderen Schriften ist, wird oft empfohlen, diese Schrift relativ zu den anderen Schriften ein wenig zu verkleinern. Dazu verwenden Sie die folgende Anweisung:

```
\usepackage[scaled=0.95]{helvet}
```

Die PostScript-Standardschriften stehen sowohl in der LaTeX-Original-Kodierung OT1 als auch in der erweiterten Kodierung T1/TS1 (wie bei den EC- und TC-Schriften) zur Verfügung. Die Dokumentation empfiehlt ausdrücklich, mit den beiden folgenden Kommandos die erweiterte Kodierung zu verwenden:

```
\usepackage[T1]{fontenc} % T1-Codierung statt OT1-Codierung
\usepackage{textcomp}    % Sonderzeichen mit TS1-Codierung
```

Wir haben allerdings die Erfahrung gemacht, dass PDF-Dokumente bei der T1-Kodierung im Adobe Reader manchmal schlechter lesbar sind als bei der Original-Kodierung OT1. Experimentieren Sie gegebenenfalls selbst!

Eine Eigenheit bei der Verwendung von PostScript-Schriften besteht darin, dass die beiden Apostrophe ' und ' nur mit einer Lupe voneinander zu unterscheiden sind. Typografisch mag das durchaus korrekt sein, wenn die Zeichen aber in Programmlistings benötigt werden, ist das ein großes Problem. Eine Notlösung besteht darin, den nach rechts gerichteten Apostroph durch das Mathematikkommando `$\grav$` zu bilden (liefert `).

Beliebige PostScript-Schriften nutzen: Während die Verwendung der PostScript-Standardschriften unkompliziert ist, erfordert die Nutzung anderer PostScript-Schriften einigen Aufwand: Die Schriften müssen so installiert werden, dass LaTeX, `dvips` und GhostScript sie finden. Hier fehlt allerdings der Platz, alle Details zu beschreiben. Stattdessen muss ich Sie auf die folgenden Internetseiten verweisen:

Kurzbeschreibung im Rahmen des Font-HOWTO-Dokuments:
http://www.tldp.org/HOWTO/Font-HOWTO/

Ausführliche, deutschsprachige Beschreibung von Christian Kuhn:
http://www.qno.de/computer/latex/fonts/tutorial.html

Ebenso ausführliche, englische Beschreibung von Matthew Amster-Burton:
http://www.mamster.net/tex/latex-fontfaq-amster-burton.pdf

8.14 LyX – LaTeX leicht gemacht

Das Programm LyX macht die Verwendung von LaTeX so einfach und komfortabel, dass selbst Einsteiger damit auf Anhieb zurechtkommen. LyX stellt eine WYSIWYG-Oberfläche zur Verfügung, in der das LaTeX-Dokument eingegeben werden kann. Echtes WYSIWYG (*What you see is what you get*) ist in Kombination mit LaTeX zwar leider unmöglich, LyX nähert sich aber erstaunlich weit an diese Idealvorstellung an.

Vielleicht wundern Sie sich, warum sich dieser Abschnitt am Ende und nicht am Anfang des LaTeX-Kapitels befindet. LyX ist sicherlich viel einfacher zu bedienen als LaTeX . Dennoch ist es für das Verständnis von LyX ungemein hilfreich, wenn Sie eine Vorstellung davon haben, wie LaTeX funktioniert.

Dieser Abschnitt basiert auf LyX 1.3.2. Von LyX existieren momentan zwei parallele Versionen, eine auf Basis der XForms-Bibliothek und eine zweite auf Basis der QT-Bibliothek. Eine dritte Variante auf Basis der Gtk-Bibliothek ist in Arbeit. Diese unterschiedlichen Versionen bieten die gleichen Funktionen,

allerdings gibt es Unterschiede, was das Aussehen der Menüs und Dialoge betrifft. Die Abbildungen zu diesem Abschnitt wurden mit der QT-Version erstellt. Weitergehende Informationen zu LyX finden Sie im teilweise leider veralteten LyX-Hilfesystem sowie auf *http://www.lyx.org/*.

8.14.1 Was ist LyX (und was ist es nicht)?

LyX ist ein moderner Texteditor. Textteile können in verschiedenen Schriftgrößen und -attributen formatiert werden und werden innerhalb von LyX auch entsprechend angezeigt (WYSIWYG). Die Bedienung des Programms ähnelt in vielen Details einem herkömmlichen Textverarbeitungsprogramm.

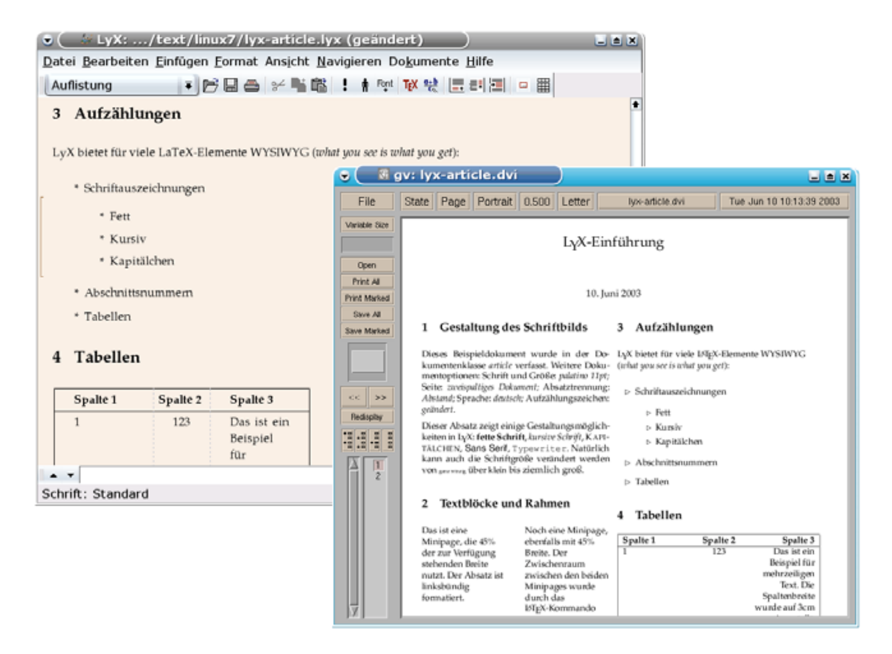


Abbildung 8.8: Das Programm LyX mit Seitenvorschau

LyX ist aber kein Textverarbeitungsprogramm im Stil von Microsoft Word oder OpenOffice Writer. Insbesondere wird die Formatierung von LyX-Dokumenten durch die Möglichkeiten von LaTeX bestimmt.

- Es ist nicht möglich, den Abstand zwischen zwei Absätzen einfach durch ein paar Leerzeilen zu vergrößern – das ist in LaTeX nicht vorgesehen.
- Ebenso ist es unmöglich, zwei Wörter durch mehrere Leerzeichen weiter voneinander zu trennen. (Sie können den Abstand zwischen zwei Wörtern vergrößern, indem Sie LaTeX-Kommandos wie `\quad`, `\qquad` oder `\hspace`

in den Text einfügen. Generell können Sie in LyX-Texten auf alle LaTeX-Kommandos zurückgreifen.)

- Auch bei der Auswahl von Schriftarten gelten die LaTeX-üblichen Einschränkungen. Es gibt lediglich eine Standardschrift sowie zwei Zusatzschriften (Sans Serif und `Typewriter`).
- LyX kennt keine Tabulatoren per Tastendruck. Wie in LaTeX müssen Sie auf *tabbing* oder Tabellen zurückgreifen.
- LyX bietet nur beinahe WYSIWYG. Unterschiedliche Schriftarten, Schriftgrößen, Tabellen etc. sind sofort zu sehen. Andere Details wie der Zeilen- und der Seitenumbruch, Seitennummern, Trennungen, Kopf- und Fußzeilen etc. werden aber erst in der Seitenvorschau sichtbar. Dazu muss das LyX-Dokument in das DVI- oder PostScript-Format übersetzt werden (*Ansicht*-Menü).

Diese Punkte stellen nur dann eine echte Einschränkung dar, wenn Sie – etwa zur Gestaltung eines Plakats – großen Wert auf ein ganz spezielles Layout legen. Dafür ist LyX schlicht ungeeignet (und LaTeX auch nur mt Tricks zu bewegen). LyX bietet sich eher zum Verfassen aller Texte an, bei denen Sie sich den Fähigkeiten von LaTeX vertrauensvoll überlassen wollen – so wie wir bei diesem Buch. Das automatisch von LyX und LaTeX erzeugte Layout sieht in den meisten Fällen auch professioneller aus als eigene Kreationen.

8.14.2 LyX-Dokumente erstellen, bearbeiten und ausdrucken

Ein neues LyX-Dokument beginnt mit dem Befehlen *Datei* → *Neu* oder *Datei* → *Neu von Vorlage*. Bei der zweiten Variante können Sie zwischen einigen LyX-Dokumentvorlagen auswählen. Leider funktionieren diese Vorlagen nur, wenn alle möglichen LaTeX-Zusatzpakete installiert sind (was per Default aber selten der Fall ist; lesen Sie gegebenenfalls *Hilfe* → *LaTeX-Konfiguration*). Die weitere Bearbeitung des Texts erfolgt im Prinzip wie bei jedem anderen Textverarbeitungsprogramm. Einige Besonderheiten von LyX werden auf den folgenden Seiten beschrieben. LyX bietet mit (Strg)+(Z) eine unbeschränkte Undo-Funktion.

Da LyX nur teilweises WYSIWYG bieten kann, muss gelegentlich kontrolliert werden, wie Details des Dokuments beim Ausdruck wirklich aussehen. LyX bietet unterschiedliche Varianten der Seitenvorschau für die Formate DVI, PostScript, PDF und HTML. Das Dokument wird für die Vorschau als LaTeX-Datei gespeichert, mit `latex` in das DVI-Format und eventuell mit `dvips` in das PostScript-Format umgewandelt. Bei umfangreichen Dokumenten dauert dieser Prozess einige Sekunden.

Tipp

Am schnellsten führen Sie die DVI- und PostScript-Vorschau mit den Tastenkürzeln Strg+D bzw. Strg+T durch. Strg+Shift+D bzw. Strg+Shift+T aktualisieren die Vorschau, ohne das Betrachtungsprogramm neu zu starten. In welchem Programm die Vorschau angezeigt wird (z. B. `xdvi` oder `gv`) kann durch *Bearbeiten* → *Einstellungen* → *Dateiformate* konfiguriert werden.

Auch für den Ausdruck (*Datei* → *Drucken*, Kürzel Strg+P) wird das LyX-Dokument in das PostScript-Format umgewandelt. Alternativ haben Sie die Möglichkeit, einen Export in die Formate ASCII, LaTeX, DVI, PostScript, PDF und HTML durchzuführen. Damit die vielen Vorschau- und Exportvarianten funktionieren, müssen natürlich die in diesem Kapitel bereits beschriebenen Programme `dvips`, `dvipdf`, `dvipdfm`, `latex2html` etc. installiert sein. Wenn Sie diese Programme nach LyX installieren, müssen Sie *Bearbeiten* → *Neu konfigurieren* ausführen und LyX neu starten, damit LyX die Programme auch findet.

Texteingabe und Sonderzeichen: Bei der Texteingabe muss keine Rücksicht auf eventuelle LaTeX-Sonderzeichen genommen werden. Anders als in LaTeX ist die Verwendung von Zeichen wie \, { oder } also problemlos; LyX kümmert sich automatisch um deren richtige Behandlung.

Einige Steuercodes (z. B. bedingte Trennzeichen) können mittels *Einfügen* → *Sonderzeichen* in den Text eingefügt werden. Zur Eingabe mathematischer Sonderzeichen (Pfeile, griechische Buchstaben etc.) verwenden Sie am besten den Dialog *Einfügen* → *Mathe* → *Mathe-Kontrollfläche* (siehe Abbildung).

Programmcode einfügen: Um Programmcode in der Typewriter-Schrift darzustellen, verwenden Sie die Absatzvorlage *LyX-Code*. Manchmal funktioniert das Kopieren von Codezeilen mit der mittleren Maustaste nicht zufriedenstellend. (Zeilenumbrüche und Einrückungen gehen verloren.) Um das zu vermeiden, verwenden Sie besser:

- *Bearbeiten* → *Externe Auswahl einfügen* → *Als Zeilen*
- *Einfügen* → *Datei einfügen* → *ASCII als Zeilen*

8.14.3 Textformatierung

Dokumenteigenschaften: Mit *Format* → *Dokument* können Sie unzählige Details einstellen, die das gesamte Dokument betreffen: Die gewünschte Dokumentvorlage, die Default-Größe der Schrift (meist 10, 11 oder 12 Punkt), den Zeichensatz des Texts, das Seitenformat, die Sprache (wichtig für die Silbentrennung) etc. LaTeX-Experten können zudem zusätzliche LaTeX-Kommandos (z. B. `\usepackage`-Anweisungen) für den LaTeX-Vorspann angeben.

Um pixelige Schriften beim PostScript- und PDF-Export zu vermeiden, sollten Sie im etwas missverständlich beschrifteten Feld *Schrift und Größe* des Dialogs *Format* → *Dokument* → *Format* eine der zur Auswahl stehenden Optionen verwenden (z. B. *pslatex*, *times* oder *palatino*).

Absatzformatierung: Zur Formatierung von Absätzen stehen eine Menge Absatzformate zur Auswahl, z. B. *Standard* für gewöhnlichen Text, *Abschnitt*, *Unterabschnitt* etc. für Überschriften sowie *Aufzählung* und *Auflistung* für Listen. LyX bietet aber leider keine einfache Möglichkeit, vorhandene Formatvorlagen zu ändern oder neue zu definieren.

Möglichkeiten zur individuellen Absatzformatierung bietet der Dialog *Format* → *Absatz*: Dort können Sie die Textausrichtung und die Abstände vor und nach dem Absatz verändern, vor und nach dem Absatz eine Linie zeichnen etc. Ob Standardabsätze durch eine Einrückung der ersten Zeile oder durch einen Abstand gekennzeichnet werden, wird durch *Format* → *Dokument* → *Format* → *Absatztrennung* eingestellt.

Wenn Sie zwei Absätze ohne Abstand, Einrückungen etc. aneinander reihen möchten, verwenden Sie (Strg)+(↵) statt einfach (↵) zur Trennung. In LaTeX entspricht das dem \\.

Mit (Alt)+(↵) zerlegen Sie einen Absatz in zwei Teile, wobei für beide neuen Absätze dasselbe Layout gilt. (Wenn Sie einfach nur (↵) verwenden, gilt für den unteren Absatz die Standardformatierung.

(Strg)+(Shift)+(C) kopiert die Formatvorlage des aktuellen Absatzes. (Strg)+(Shift)+(V) wendet dieses Format anschließend auf einen anderen Absatz ein. Leider werden durch diese Tastenkürzel keine Zeichenformate und nur ein Teil der individuellen Absatzformate kopiert.

Zeichenformatierung: *Format* → *Zeichen* verändert die Schriftfamilie, -größe und -form des zuvor markierten Texts. Die Option *Alle umschalten* bewirkt, dass die Formatoptionen bei einem mehrfachen Anklicken von *Übernehmen* ein- und wieder ausgeschaltet werden. Verwenden Sie die Tastenkürzel (Strg)+(B) für fette Schrift (*bold*), (Strg)+(E) für kursive Schrift (*emphasized*) sowie (Strg)+(Shift)+(P) für Typewriter. Eine nochmalige Anwendung des Kommandos macht die Formatierung rückgängig. (Alt)+(Z), (Leertaste) entfernt alle Zeichenformate.

Die *Format*-Dialoge können während der Texteingabe ständig geöffnet bleiben. Es ist nicht notwendig, die Dialoge nach jeder Einstellung wieder zu schließen.

Aufzählungen: Listen bilden Sie mit der Formatvorlage *Auflistung* oder *Aufzählung* (nummeriert). Zur Verschachtelung von Listen (Unterpunkte etc.) rücken Sie die betreffenden Einträge durch *Format* → *Umgebungstiefe erhöhen* ein. Die umgekehrte Wirkung hat das Kommando - → *Umgebungstiefe ver-*

ringern. Beide Kommandos können auch per Tastatur ausgeführt werden: Shift+Alt+→ bzw. Shift+Alt+←. Das Aussehen der Aufzählungspunkte können Sie im Dialog *Format → Dokument → Aufzählungszeichen* ändern.

8.14.4 Besondere Textelemente (Tabellen, Fußnoten, Formeln)

Tabellen: *Einfügen → Tabelle* erstellt eine Tabelle mit einer beliebigen Anzahl von Zeilen und Spalten. In den einzelnen Zellen können Sie nun Text eingeben. Zur Formatierung der Tabelle klicken Sie diese mit der rechten Maustaste an. Im nun erscheinenden Dialog können Sie Zeilen und Spalten hinzufügen bzw. löschen. Das Feld *Horizontale Ausrichtung* verändert die Ausrichtung der gerade aktuellen Spalte (linksbündig, rechtsbündig oder zentriert) bzw. aller markierten Spalten. Im Dialogblatt *Rahmen* können Sie für die markierten Zellen die Rahmenlinien löschen oder setzen. Die Linien reichen immer über die gesamte Breite bzw. Höhe der Tabelle.

Die Tabelle als Ganzes gilt als Absatz. Wenn Sie die Tabelle also zentrieren oder die Abstände vorher oder nachher verändern möchten, müssen Sie *Format → Absatz* ausführen.

Per Default ergibt sich die Breite der Spalten jeweils aus dem längsten Eintrag. Nur wenn Sie die Spaltenbreite exakt einstellen (z. B. auf 5 cm), ist innerhalb eines Tabellenfelds auch mehrzeiliger Text erlaubt. Allerdings scheint LyX eine nachträgliche Änderung der Spaltenbreite nicht immer zu akzeptieren.

Umfangreiche Tabellen sollten als Gleitobjekte erstellt werden. Dazu führen Sie zuerst *Einfügen → Gleitobjekte → Tabelle* aus, bewegen den Cursor an den Anfang oder an das Ende des Gleitobjekts und führen dort *Einfügen → Tabelle* aus. LyX platziert die Tabelle samt Beschriftung automatisch auf der nächsten Seite, wenn sie auf der aktuellen Seite keinen Platz findet. Gleitende Tabellen haben allerdings den Nachteil, dass die Tabellenbeschriftung nur zentriert erfolgen kann.

Fußnoten und Randnotizen: Fußnoten und Randnotizen werden mit *Einfügen → Fußnote* bzw. *Einfügen → Randnotiz* gebildet. LyX fügt damit einen Button in den Text ein. Den Text der Fußnote bzw. der Randnotiz können Sie durch Anklicken des Buttons lesen bzw. verändern. Erst bei der Seitenvorschau werden diese Objekte am richtigen Ort angezeigt.

Querverweise: Um Querverweise nutzen zu können, müssen Sie zuerst mit *Einfügen → Marke* eine Textmarke setzen. Diese Marke kann einen beliebigen Namen haben. Anschließend können Sie an einer anderen Stelle im Text darauf verweisen: Dazu führen Sie *Einfügen → Querverweis* aus, wählen die gewünschte Textmarke aus und geben das gewünschte Format an (<*Referenz*> für die Abschnittsnummer, <*Seite*> für die Seitennummer).

Abbildungen: *Einfügen → Grafik* führt zu einem Dialog, um eine Grafikdatei in das Dokument einzubinden. Sie können dort den Dateinamen angeben, die

Grafik auf die gewünschte Größe skalieren etc. Anders als LaTeX unterstützt LyX auch diverse Bitmap-Formate (z. B. JPEG oder PNG). LyX wandelt derartige Bilder vor der LaTeX-Übersetzung automatisch in EPS-Dateien um. Das funktioniert freilich nur, wenn entsprechende Umwandlungsprogramme installiert und von LyX auch erkannt werden (siehe *Bearbeiten* → *Einstellungen* → *Konverter*). Per Default ist das oft nicht der Fall, weswegen auch unter LyX eine Beschränkung auf PostScript- bzw. EPS-Dateien sinnvoll ist.

Um Abbildungen zu beschriften, werden diese als Gleitobjekte eingefügt: Führen Sie zuerst *Einfügen* → *Gleitobjekte* → *Grafik* aus und fügen Sie dann die Grafik mit *Einfügen* → *Grafik* ein. Je nachdem, wo Sie die Grafik einfügen, erfolgt die Beschriftung ober- oder unterhalb. Die Grafik gilt als eigener Absatz. Da die Beschriftung zentriert erfolgt, sollte auch die Grafik zentriert werden (*Format* → *Absatz*).

Querverweise auf Bilder werden wie gewöhnliche Querverweise gebildet (siehe oben). Die Definition der Textmarke muss in der Beschriftungszeile des Gleitobjekts erfolgen.

8.14.5 Mathematische Formeln

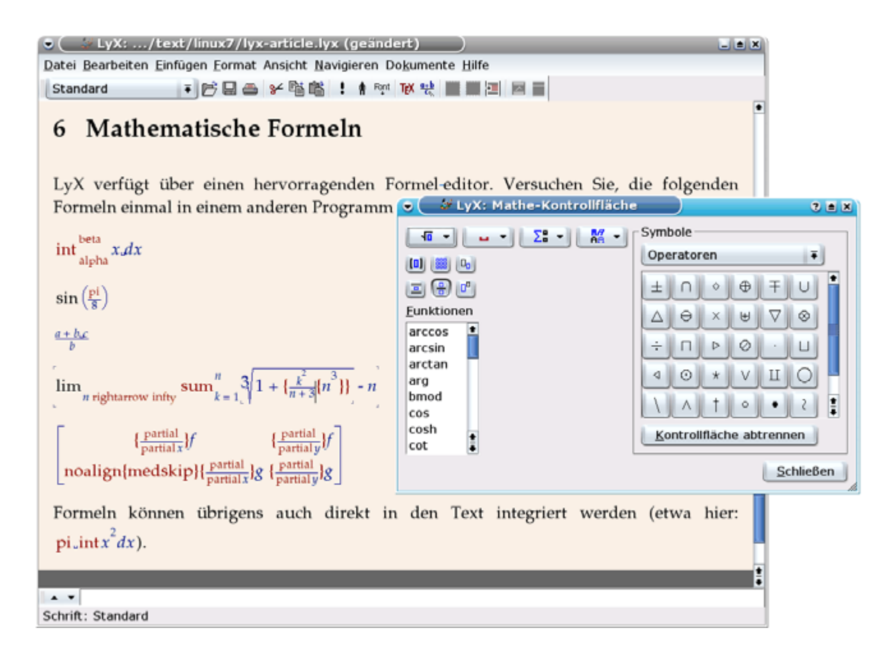


Abbildung 8.9: Formeleingabe in LyX

Die Eingabe mathematischer Formeln beginnt mit Strg+M oder Strg+Shift+M. Beide Kommandos führen in den Mathematik-Modus,

der mit Esc wieder verlassen wird. Der Unterschied liegt in der Darstellung der Formel: Im ersten Fall wird die Formel in den Fließtext integriert, im zweiten Fall gilt sie als eigenständige Formel (eigener Absatz, zentrierte Darstellung). Bei bereits bestehenden Formeln kann der Darstellungs-Modus ebenfalls mit Strg+Shift+M umgeschaltet werden.

Bei der Eingabe von Formeln hilft die *Mathe-Kontrollfläche*, die nach einem Klick mit der rechten Maustaste auf die Formel erscheint (siehe Abbildung **??**). Außerdem können fast alle Formelelemente sehr einfach per Tastatur eingegeben werden. LaTeX-Kenner werden es begrüßen, dass die meisten LaTeX-Kommandos direkt eingegeben werden dürfen (z. B. *frac*, Leertaste zur Eingabe eines Bruchs). Leider werden die meisten mathematischen Sonderzeichen nicht durch entsprechende Symbole, sondern als Text dargestellt. (Ältere LyX-Versionen boten in dieser Hinsicht mehr WYSIWYG.)

LyX passt die Größe von Klammern automatisch an den Inhalt der darin enthaltenen Elemente (z. B. einer Matrix) an. Das funktioniert allerdings nur dann, wenn die Klammern entweder mit Alt+M, (bzw. mit Alt+M, [eingegeben werden.

Tastenkürzel zur Eingabe mathematischer Formeln

Strg+M	wechselt in den Mathematik-Modus
Strg+Shift+M	wechselt zwischen abgesetzten und eingebetteten Formeln
Esc	verlässt den Mathematik-Modus, setzt den Cursor an das Ende der Formel
Leertaste	mathematisches Element (z. B. Bruch, Klammernebene) verlassen
Strg+Leertaste	kleiner Abstand zwischen zwei mathematischen Elementen
\code Leertaste	ersetzt \code (siehe ab Seite 382) durch das entsprechende Symbol
_	tiefstellen
^	hochstellen
Alt+M, 8	Unendlich-Symbol (∞)
Alt+M, (oder [etc.	Klammernpaar einfügen
Alt+M, =	Ungleichheitszeichen ($\neq$)
Alt+M, +	Plus/Minus ($\pm$)
Alt+M, F	Bruch (*frac*)
Alt+M, G, Buchstabe	griechische Buchstaben (z. B. Alt+M, G, B für β)
Alt+M, I	Integral
Alt+M, P	Symbol für partielle Ableitung (∂)
Alt+M, S	Wurzel (*sqrt*)
Alt+M, V	Vektorpfeil über das nächste Zeichen ($\vec{v}$)

Leerzeichen sind in Formeln nicht vorgesehen, denn LaTeX kümmert sich selbst um die richtigen Abstände. Daher bewirkt (Leertaste) in LyX keine Eingabe, sondern das Verlassen der gerade aktiven Formelebene. Wenn Sie einen kleinen Abstand zwischen zwei Symbolen eingeben möchten (etwa um bei einer Multiplikation klar zu machen, dass es sich bei $a\,b$ um das Produkt zweier Variablen und nicht um die Variable ab handelt), verwenden Sie dazu einfach (Strg)+(Leertaste). Zur Feineinstellung des Layouts stehen darüber hinaus auch andere Abstände zur Verfügung. Solche Abstände werden mit *Einfügen → Mathe* oder über die *Mathe-Kontrollfläche* gebildet.

8.14.6 LyX-Besonderheiten

Navigation in langen Texten: Das Menü *Navigieren* hilft dabei, den Cursor rasch an den Beginn eines beliebigen Abschnitts des Texts zu bewegen. Bei umfangreichen Dokumenten können Sie außerdem mit *Ansicht → Inhaltsverzeichnis* einen Dialog öffnen, der alle Abschnitte und Unterabschnitte des aktuellen Dokuments enthält.Die Auswahl eines Elements in diesem Strukturdialog bewegt den Cursor an die entsprechende Position.

LaTeX-Kommandos: Profis können LaTeX-Kommandos mit *Einfügen → TeX* bzw. mit (Strg)+(L) direkt in den Text einfügen. Der Code wird im Text als Button angezeigt und erst beim Druck des Dokuments bzw. bei der Seitenansicht ausgeführt.

LaTeX-Import und -Export: Mit *Datei → Importieren* bzw. *Datei → Exportieren* können Sie LaTeX-Dokumente importieren bzw. exportieren. Der Importfilter funktioniert nur dann akzeptabel, wenn im LaTeX-Text weder eigene Makros noch spezielle LaTeX-Pakete eingesetzt wurden. Vollkommen problemlos ist hingegen der LaTeX-Export – LyX ist ja nichts anderes als eine komfortable Benutzeroberfläche zu LaTeX.

Bildschirmdarstellung: LyX verwendet drei TrueType- oder PostScript-Schriften zur Darstellung des Texts am Bildschirm. Diese Schriften und ihre Grundgröße können Sie einstellen – ebenso einen globalen Zoom-Faktor. Die Bildschirmschriften gelten nur für die Anzeige innerhalb von LyX. Welche Schriften in der Vorschau bzw. beim Ausdruck verwendet werden, hängt von *Format → Dokument → Schrift und Größe* ab.

Interna: LyX ist auf zahlreiche externe Programme angewiesen, insbesondere auf viele teTeX-Komponenten. Bei der Installation überprüft LyX automatisch, welche Programme verfügbar sind. Das Ergebnis dieser Überprüfung können Sie mit *Hilfe → LaTeX-Konfiguration* testen.

Wenn Sie später weitere externe Programme aus dem Umfeld von teTeX, LaTeX oder LyX installieren, müssen Sie in LyX *Bearbeiten → Neu konfigurieren* ausführen. Die neu installierten Komponenten stehen nach einem LyX-Neustart zur Verfügung.

8.15 Aufgaben

1. Schreiben Sie ein Muster-Präambel für einem Artikel, den Sie mit LaTeX verfassen wollen.
2. Erstellen Sie den LaTeX-Sourcecode für folgende Auflistung:

   Checkliste für mehrtägige Exkursionen

   - Waschzeug
     - Handtuch
     - Zahnbürste
     - Zahnpasta
     - Kamm/Bürste
     - Seife/Duschgel
   - Hausschuhe
   - Kleidung (genug)
   - Unterwäsche (ebenfalls genug)
   - Badehose
   - Trinkflasche
   - Schlafanzug
   - Ohrstöpsel
   - Wärmflasche (für Weicheier)
   - Notfallausrüstung
     - Handy
     - Verbandszeug
     - Alka Selzer
     - Rettungsdecke
     - MP3-Player
     - Taschenmesser
3. Setzen Sie folgende Tabelle mit LaTeX:

Nahrungsmittel	Art	Ausgangsprodukt
Schnitzel	Fleisch	Schwein
Steak	Fleisch	Rind
Semmel	Getreide	Weizen
Pop Corn	Getreide	Mais
Joghurt	Milchprodukt	Kuhmilch
Knödel	Gemüse	Kartoffel

4. Setzen Sie folgende Formel mit LaTeX:

   Mit $k = \sqrt[3]{8}$:

$$k^2 \in \left\{ \sum_{i=1}^{\infty} \frac{1}{2^i}, \ldots, \sum_{i=1}^{\infty} \frac{1}{i} \right\}$$

5. Versuchen Sie, folgende kleine Mathelektion zu setzen:

   Jeder Mathematiker weiß, dass z. B. die Summe von zwei Größen nicht etwa in der Form

$$1 + 1 = 2 \tag{8.2}$$

   dargestellt wird. Diese Form ist viel zu schlicht. Schon Anfangssemester wissen, dass gilt:

$$1 = \ln e \tag{8.3}$$

   weiterhin ist geläufig, dass

$$1 = \sin^2 q + \cos^2 q \ . \tag{8.4}$$

   Ausserdem ist dem kundigen Leser offensichtlich, dass

$$2 = \sum_{n=0}^{\infty} \frac{1}{2^n} \ . \tag{8.5}$$

   Daher kann die Gleichung (C.1) viel wissenschaftlicher in der Form

$$\ln e + (\sin^2 q + \cos^2 q) = \sum_{n=0}^{\infty} \frac{1}{2^n} \ . \tag{8.6}$$

   ausgedrückt werden.

Kapitel 9

Anwendungsprogramme

Einführung

Sie haben bis hierher die Shell und auch etliche andere Programme kennengelernt – zum Teil kleine Tools, aber auch Anwendungen für spezielle Zwecke wie die Scan- und Bildbetrachtungsprogramme. Nicht zuletzt haben wir Sie mit dem Textsatz vertraut gemacht.

Dieses Kapitel beschreibt eine Auswahl von Anwendungsprogrammen, die wir aus der schier endlosen Zahl von Linux-Programmen herausgepickt haben. Die ausgewählten Anwendungen stammen aus den verschiedensten Bereichen und sollen Sie dazu anregen, auch mal gezielt nach einer Linux-Anwendung für einen bestimmten Zweck zu suchen. Sie werden überrascht sein, was es da alles gibt. Allein der ISIS-Report für Linux-Software listete im Jahr 2005 mehr als 2000 Linux-Lösungen auf. Wir haben auch nach Programmen gesucht, die nicht bei jeder Distribution schon im Menü auftauchen.

Natürlich zwingt uns der Umfang des Buchs auch in diesem Kapitel wieder zur Bescheidenheit. Daher werden wir die Programme in der Regel nur vorstellen; eine Beschreibung im vollen Umfang der Möglichkeiten eines Programms würde da jeden Raum sprengen – denken Sie nur an so Tausendsassas wie OpenOffice oder Gimp.

Verachten Sie bitte auch die kleinen Tools nicht (die in diesem Kapitel nicht vorkommen). Gerade diese Progrämmchen sind oftmals hilfreich und lassen sich per Pipe oder Shell-Script zu mächtigeren Werkzeugen zusammensetzen.

So ist die folgende Auswahl ebenso subjektiv wie absolut unvollständig. Wir hoffen aber, möglicherweise den einen oder anderen Aha-Effekt bei Ihnen ausgelöst zu haben.

9.1 Home Office

9.1.1 OpenOffice

OpenOffice ist das momentan populärste und leistungsfähigste frei verfügbare Office-Paket für Linux. Die wichtigsten Vorteile sind:

- Relativ hohe Kompatibilität mit Microsoft Office: Das betrifft sowohl die Bedienung als auch den Import und Export von Dateien.
- Unterstützung für die Betriebssysteme Linux, Windows und Sun Solaris: Damit ist ein unkomplizierter Dokumentenaustausch in heterogenen Umgebungen möglich. (Eine Mac-OS-X-Version ist geplant, wird aber noch einige Zeit auf sich warten lassen.)
- Umfassende Funktionalität, auch für anspruchsvolle Aufgaben geeignet (z. B. zum Verfassen einer Magisterarbeit)
- Ausgereift, stabil und sicher

Dieser Abschnitt beschreibt in wenigen Worten die Eigenschaften von OpenOffice, denn gerade zu diesem Programm gibt es schon viele Bücher und Dokumentationen. Wer trotz des vorhergehenden Kapitels seine Texte lieber mit einem WYSIWYG-Programm erstellt, ist mit OpenOffice bestens bedient. Das Paket aus Textverarbeitung, Tabellenkalkulation, Präsentation und Datenbank hat alles Wichtige für das Büroleben integriert.

Es gibt unterschiedliche Meinungen dazu, ob OpenOffice nun ein vollwertiger Ersatz für MS-Office ist oder nicht. Für die Gelegenheitsanwenderin bietet OpenOffice sicherlich mehr als genug Funktionen. Auch der Umstieg sollte kein Problem sein – die Menükommandos beider Office-Pakete sind ganz ähnlich strukturiert. Wenn Sie aber zum Kreis der so genannten Power-Userinnen zählen, müssen Sie sich auf manche Einschränkungen gefasst machen.

- Die Import- und Export-Funktionen von OpenOffice reichen für einen problemlosen Dokumentenaustausch mit den Microsoft-Produkten noch immer nicht aus. Der Import eines Dokuments nach OpenOffice gelingt fast immer. Oft ist danach jedoch Handarbeit erforderlich, bis alles wieder so aussieht, wie es soll. Anschließend sollte das Dokument aber unbedingt in einem OpenOffice-Format gespeichert werden und in diesem Format bleiben. Vom wiederholten Import und Re-Export in ein Microsoft-Format ist abzuraten.
- OpenOffice ist nicht in der Lage, passwort-geschützte MS-Office-Dateien zu öffnen.

- Manche OpenOffice-Funktionen sind deutlich langsamer als unter MS-Office. Dazu zählen insbesondere das Laden und Speichern umfangreicher Dokumente. Erschwerend kommt hinzu, dass MS-Office manche zeitaufwändige Operationen im Hintergrund durchführen kann, während OpenOffice in dieser Zeit vollständig blockiert ist.
- Viele Tastenkürzel haben eine andere Bedeutung als in MS-Office. Es besteht aber immerhin die Möglichkeit, vorhandene Kürzel zu ändern bzw. eigene Kürzel zu definieren. Für Vielschreiberinnen gibt es leider eine wesentliche Einschränkung: Formatvorlagen kann kein Tastenkürzel zugewiesen werden.
- OpenOffice bietet keine Komponente an, die mit Access vergleichbar ist. Sie können zwar auf Datenbanken zugreifen und Daten importieren, verändern und geordnet ausdrucken, das Funktionsspektrum von Access wird aber nur zu einem kleinen Bruchteil abgedeckt.
- OpenOffice unterstützt VBA nicht. (VBA ist die Makroprogrammiersprache des MS-Office-Pakets.) VBA-Code bleibt beim Import/Export zwar normalerweise erhalten, kann aber nicht ausgeführt werden. Vorsicht: Beim Import/Re-Export von Excel-Dateien werden alle Formeln zerstört, die auf selbst programmierte VBA-Funktionen verweisen. OpenOffice bietet als Alternative zu VBA eigene Programmiermöglichkeiten an. Diese sind aber inkompatibel zu VBA, bieten weniger Funktionen und einen deutlich geringeren Komfort bei der Programmierung.

Kurz vor Redaktionsschluss dieses Buchs hat das OpenOffice-Projekt die offizielle Version OpenOffice 2.0 vorgelegt. Insgesamt ähneln die einzelnem Komponenten nun den Windows-Konkurrenten sehr viel stärker, was den Umstieg wesentlich erleichtern wird. Jedoch ist die Verarbeitung von Windows-Scripting (gottseidank?) noch immer nicht vorgesehen. Auch ist die Interoperabilität vor allem mit Microsoft Office stark verbessert worden; Anwender, die von Microsofts Office-Paket wechseln wollen, werden sich nun in OpenOffice besser zurecht finden.

Eine der wichtigsten Neuerungen in OpenOffice 2.0 dürfte das XML-Dateiformat sein, das den Standards der Organization for the Advancement of Structured Information Standards (OASIS) entspricht. Die Organisation hat in OpenDocument Definitionen festgelegt, mit der sich Dateien aus Office-Paketen zwischen unterschiedlichen Softwaresystemen und Plattformen problemlos austauschen lassen sollen.

Ausgebaut wurden auch die schwächlichen Datenbankfähigkeiten von OpenOffice; zu den Komponenten Textverarbeitung, Tabellenkalkulation, Präsentations- und Grafiksoftware kommt in der Version 2 die Datenbank „Base“ hinzu, die auch als Frontend für den Zugriff auf SQL-, ODBC-, dBase- und andere Datenbanken dient. Das neue Frontend soll den Umgang mit den Datenbankfähigkeiten erleichtern. So lassen sich nun etwa neue Datenbanken einfach

per Datei-Menü anlegen. Ein Tabellenassistent soll zudem bei der Erstellung von Datenbanken Hilfestellung leisten.

Wie immer finden Sie im Internet jede Menge weiterführende Dokumentation. Die folgenden Links können ein erster Startpunkt bei der Suche sein:

http://www.OpenOffice.org bzw. *http://de.OpenOffice.org*
http://de.OpenOffice.org/about-documentation.html
http://de.OpenOffice.org/doc/faq/mainfaq/index.html
http://www.bytebot.net/OpenOffice/faq.html
http://support.OpenOffice.org/
http://www.oooforum.org/
http://ooodocs.sourceforge.net/
http://OpenOffice.start4all.com/

9.1.2 OpenOffice-Alternativen

Sun StarOffice: OpenOffice ist aus StarOffice entstanden. StarOffice ist ein Office-Paket, das vor vielen Jahren in Deutschland von der Firma Star Division entwickelt wurde. 1999 kaufte Sun StarOffice und machte in der Folge den Quellcode des Office-Pakets öffentlich verfügbar. Seither arbeiten sowohl Mitarbeiter von Sun als auch zahlreiche freie Programmierer an der Weiterentwicklung.

Daher gibt es zwei Varianten des Office-Pakets: die Open-Source-Variante OpenOffice (Lizenz LGPL) und die kommerzielle Variante StarOffice. Beide Versionen sind weitestgehend kompatibel zueinander, StarOffice bietet aber einige kommerzielle Zusatzkomponenten, unter anderem eine bessere Rechtschreibprüfung, zusätzliche Fonts, Clipart etc. Außerdem ist zu StarOffice traditioneller Support verfügbar, wie ihn manche Firmen wünschen. Diese Extras gibt es natürlich nicht umsonst: StarOffice ist ein kommerzielles Produkt, das käuflich erworben werden muss. Auf der StarOffice-Website (*http://wwws.sun.com/software/star/staroffice/index.html*) gibt es aber auch eine Evaluationsversion zum kostenlosen Download.

KOffice: KOffice (*http://www.koffice.org/*) ist das Office-Paket zum KDE-Desktop. Die wichtigsten Komponenten – darunter `kword`, `kspread` und `kpresenter` – sind relativ gut ausgereift. Daneben gibt es zahlreiche weitere Komponenten, z. B. Zeichenprogramme und Datenbankanwendungen, die sich allerdings zum Teil noch in einem relativ frühen Entwicklungsstadium befinden.

Generell ist die Kompatibilität mit Microsoft Office geringer als mit OpenOffice. Das betrifft nicht nur den Import und Export von Dateien, sondern auch das gesamte Bedienungskonzept. Man merkt KOffice an, dass es konzeptionell eine vollkommene Neuentwicklung ist (was durchaus mit vielen Vorteilen verbunden ist).

Der größte Vorteil von KOffice ist die nahtlose Integration in den KDE-Desktop. Bemerkenswert ist auch die gute Zusammenarbeit der Komponenten, die beispielsweise beim Einbetten unterschiedlicher Dokumenttypen spürbar wird. Im Vergleich zu OpenOffice basieren die KOffice-Komponenten auf deutlich kleineren Programmen, die entsprechend schneller gestartet werden und geringere Hardware-Anforderungen stellen. Zukünftige KOffice-Versionen werden vermutlich dasselbe Dateiformat wie OpenOffice nutzen, was den Dokumentenaustausch zwischen diesen beiden Office-Paketen sicherlich erleichtern wird.

Gnome Office: Im Gegensatz zu OpenOffice und KOffice handelt es sich bei den Gnome-Office-Komponenten um Einzelprogramme, die wenig miteinander zu tun haben. Es gibt weder ein einheitliches Dateiformat noch ein durchgängiges Konzept für die Bedienung. Die bekannteste Office-Komponente ist Gimp (siehe Kapitel 9.4.1). Sehr ausgereift ist auch das Tabellenkalkulationsprogramm `gnumeric`. Es zeichnet sich durch eine hohe Kompatibilität mit Microsoft Excel aus und bietet darüber hinaus eine Menge finanzmathematischer Zusatzfunktionen an. Ebenfalls recht brauchbar ist das Textverarbeitungsprogramm `abiword`. Die restlichen Komponenten befinden sich dagegen teilweise noch in einem frühen Entwicklungsstadium (*http://www.gnome.org/gnome-office/*).

CrossOver Office: Trotz der vielen Linux-Office-Pakete gibt es Anwenderinnen, die auf das originale Microsoft Office nicht verzichten können oder möchten. Die wahrscheinlich komfortabelste Lösung bietet in diesem Fall die Firma CodeWeaver mit dem kommerziellen Zusatzprodukt CrossOver Office (*http://www.codeweavers.com/*). Dabei handelt es sich nicht um ein eigenes Office-Paket, sondern um eine Version des Windows-Emulators WINE, die so weit verbessert wurde, dass die meisten Komponenten von Microsoft Office 97, 2000 und XP installiert und verwendet werden können. Diese Lösung ist natürlich relativ teuer – Sie benötigen eine Microsoft-Office-Lizenz und CrossOver Office.

9.1.3 AbiWord

AbiWord ist eine Open-Source-Textverarbeitung, mit der Sie alle Arten von Textdokumenten erstellen können. Die Bedienoberfläche orientiert sich stark an Microsoft Word, was einen Umstieg erleichtert. Es bietet nahezu alle wichtigen Funktionen, die man von einer professionellen Textverarbeitung erwartet.

Das schlanke Schreibprogramm ist leicht erlernbar und beschränkt sich auf die Grundfunktionen der Textverarbeitung inklusive Fonts, Linealen, Tabulatoren, Formatvorlagen, Tabellen, Fußnoten und automatischem Inhaltsverzeichnis. AbiWord bietet somit alle wesentlichen Funktionen einer Textverarbeitung und ist plattformunabhängig. Es ermöglicht einfachen Datenaustausch mit anderen Office-Anwendungen und ist vor allem durch seine Erweiterbarkeit mittels Plugins geprägt. Außerdem spricht AbiWord nicht nur deutsch, sondern auch andere Sprachen.

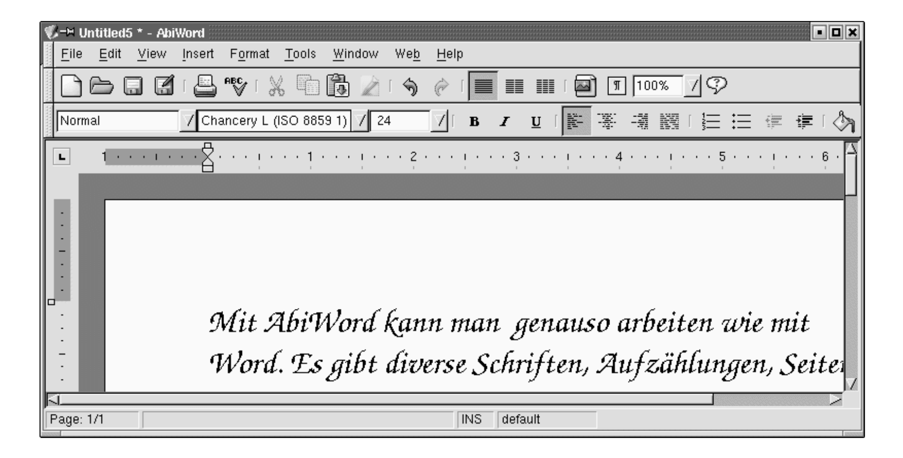


Abbildung 9.1: Leichtgewichtige Textverarbeitung mit AbiWord

AbiWord verfügt über viele Import/Export-Dateiformate wie RTF, MS Word, OpenOffice, HTML, LaTeX und sogar das nroff-Format der Manual-Seiten. Das programmeigene Dokumentformat ist ein XML-Dokumenttyp. Dokumente, die mit AbiWord geschrieben wurden, sind daher in punkto Archivierung sicher angelegt. Außerdem besitzt AbiWord einen rudimentären Import-Filter für die OpenDocument-Formate. Ein anderes nennenswertes Feature von AbiWord ist die Möglichkeit, WordPerfect-Dateien zu öffnen. Das kann sonst kaum eine der unter Linux verfügbaren Textverarbeitungen.

Neben einer umfangreichen Bearbeitung von Schrift und Layout (auch zweispaltiger Satz), können auch Bilder und Dateien eingebunden werden. Auf der Herstellerseite findet sich ein Link, um noch ein deutsches Wörterbuch und einige weitere interessante Plug-Ins herunterzuladen. Das Wörterbuch muss einfach nur in das Dictionary-Verzeichnis entpackt werden.

Der Eingabe von Text folgt AbiWord ausreichend schnell mit der Darstellung auf dem Bildschirm. Um auch bei umfangreichen Texten den Überblick zu behalten, können im Text unsichtbare Markierungen (Lesezeichen) eingefügt werden. Diese werden nicht gedruckt und dienen lediglich dem Autor als „Fixpunkte“.

Weitere Features im Überblick:
- Unbegrenztes Undo/Redo
- Suchen und Ersetzen
- Einfügen von Tabellen, Listen, Bildern, Fußnoten und Styles in Dokumente
- Multilinguale Bedieneroberfläche
- Rechtschreibprüfung in über 30 Sprachen
- Unterstützung von Rechts-nach-Links und gemischten Schreibweisen
- Integrierte Templates
- Echtzeit-Grammatiküberprüfung (Englisch)
- Fließtext um transparente Bildbereiche

– Gleichungs-Bearbeitung
– Gnome Office-Charts
– Lesezeichen im Text

Als Grundlage diente den Programmierern von AbiWord das Toolkit gtk+, welches auch bei Gnome, Gimp und vielen anderen Anwendungen benutzt wird. AbiWord ist unter Linux, Mac OS X, Microsoft Windows, aber auch anderen Betriebssystemen verfügbar. Weitere Informationen finden Sie unter *http://www.abisource.com/* und *http://www.abiword.org/*.

9.1.4 DTP mit Scribus

Einst ein Thema in jeder Computerzeitschrift, ist DTP (DeskTop Publishing) heutzutage nur noch ein Thema für Spezialisten, die wissen, dass ein wahlloses Verstreuen von Word-Textboxen auf einer Seite noch kein Zeitschriftenlayout darstellt. Ein DTP-Programm unterscheidet sich von einer Textverarbeitung durch die Layoutfunktionen. Aus den ersten Tools wie Pagemaker sind leistungsfähige Programme wie Illustrator, QuarkXpress oder FreeHand entstanden, die auch einen entsprechenden Preis haben. Wer aber nur alle Jubeljahre einmal ein derartiges Programm nutzt und auch keine ausgeprägten Profifunktionen benötigt, kann mit Scribus schöne Ergebnisse erzielen.

Von der Textverarbeitung bis zum Satz für den Druck bietet die Open-Source-Software vielfältige Möglichkeiten. Die Version 1.3.2 (Januar 2006) namens „Egalité“ ist nicht nur für Linux verfügbar, sondern auch für die Betriebssysteme Windows und OS-X, was die plattformübergreifende Arbeit erleichtert. Eine ins Deutsche übersetzte Anleitung ist auch erhältlich.

Das Programm kann ebenfalls verwendet werden, um in vorgedruckte Formulare an der richtigen Stelle Text-, Bild- oder andere Objekte einzusetzen und auf dem Formularvordruck auszugeben. Das geht deutlich besser als mit einem Textverarbeitungsprogramm.

Scribus unterstützt eine Vielzahl von Funktionen, die für professionelle Druckvorlagen benötigt werden. Dazu zählen ICC-Farbmanagement, Farbseparation durch CMYK-Ausgabe, PDFs nach PDF/X-3 Standard, PDF-Import, Tabellen und Vektorzeichnungen. Scribus bietet dabei auch erweiterte PDF-Funktionaliäten wie Erstellung von PDF-Präsentationen und PDF-Formularen. Kein Problem sind für Scribus außerdem der Import und Export von EPS-Dateien und Text in Unicode. Darüber hinaus liefert die Software Werkzeuge zum Erstellen von Vektorgrafiken, sie ermöglicht das Importieren von SVG-Grafiken und unterstützt OpenType-Fonts.

Bei DTP arbeitet man anders als bei einer Textverarbeitung, denn es geht hier um die Gestaltung einer Visitenkarte, eines Werbeflyers, einer Broschüre oder

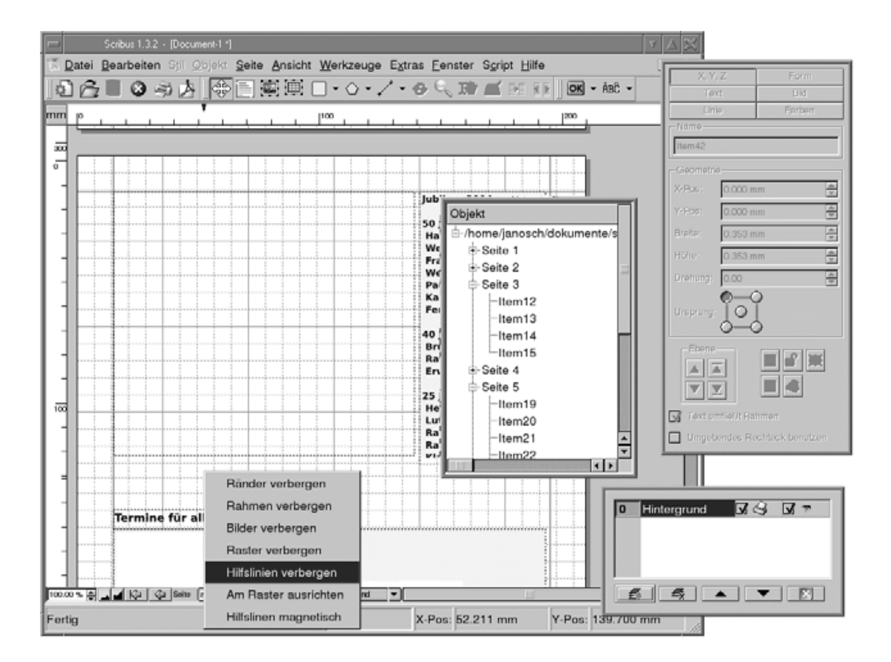


Abbildung 9.2: DTP bietet Linux mit Scribus

einer Vereinszeitschrift. Zunächst bestimmt man in der so genannten Dokumentstruktur die Formate, Seitenränder, Schriftarten usw. Auf der daraus entstehenden Satzfläche platziert man die eigentlichen Inhalte (früher klebten die Setzer mit Text oder Bildern belichtete Filme zu einer Seite zusammen, heute geschieht das am Computer). Um Texte und Bilder zu positionieren, muss man einen entsprechenden Text- oder Bildrahmen einbauen und über das Kontextmenü dieses Elementes die gewünschte Quelldatei laden.

Scribus zeigt die Seite 1:1 an und erlaubt es dann, Textrahmen, Bilder, Tabellen oder grafische Formen auf der Arbeitsfläche zu platzieren. Sie lassen sich mit der Maus immer wieder neu anordnen, um auf diese Weise das Layout zu erstellen. Alle Objekte können fast beliebig manipuliert werden. So ist es kein Problem, Bilder zu drehen, sie in mehreren Ebenen übereinander anzuordnen oder sie so zu bearbeiten, dass Texte automatisch um sie herumfließen. Texte können Sie ebenfalls umfassend formatieren. Um das Platzieren der Rahmen zu erleichtern, kann man ein Raster einblenden lassen, an dem man die Rahmen ausrichten kann. Wer seinen Text oder Bilder nicht immer rechteckig angezeigt haben möchte, kann bei Scribus auch die Rahmenformen verändern, z. B. die Ecken abrunden oder den Rahmen zu einem Oval formen. Mehrere Textrahmen lassen sich auch miteinander verknüpfen, so dass ihre Inhalte automatisch aus dem einen Rahmen hinaus- und in den nächsten wieder hineinlaufen. Interessant

ist, dass das Programm sogar Blindtexte in verschiedenen Sprachen anbietet, um auf diese Weise Probe-Layouts zu erstellen.

Um Text in den Rahmen einzufügen, klickt man in der Werkzeugleiste auf *Rahmeninhalt bearbeiten*. Nun kann man in den Rahmen schreiben. Das geht etwas umständlich, besser ist es, gleich eine bereits bestehende Textdatei einzufügen, indem man mit der linken Maustaste in den Rahmen klickt und im erscheinenden Menü *Text laden* auswählt. Diesen Text kann man dann leicht mit der Maßpalette oder dem Menü *Stil* formatieren. Wenn der Inhalt der Textdatei nicht in den Rahmen passt, erscheint unten rechts im Rahmen ein kleines X. Wollen Sie den Rahmen nicht vergrößern, kann man ihn auch mit einem anderen Rahmen verbinden. Für Bilder erstellt man über *Bildrahmen einfügen* in der Werkzeugleiste einen Rahmen. Mit Linksklick in den Rahmen öffnet man ein Menü, in dem man *Bild laden* auswählen kann. Nun sucht man sich das einzufügende Bild aus. So fahren Sie fort, bis alle Elemente im Dokument enthalten sind.

Bei regelmäßig erscheinenden Publikationen ist es natürlich nicht ratsam, jedesmal das Layout von Neuem zusammenzubauen (wie leicht wird etwas vergessen). Wie bei DTP-Software üblich, unterscheidet auch Scribus normale Seiten und so genannte Masterseiten, welche die auf allen Seiten des gesetzten Dokuments erscheinenden Elemente, z. B. Seitenzahlen oder Logos in den Unterzeilen enthalten. Auf den normalen Seiten werden dann die unterschiedlichen Artikel eingebunden. Alle Objekte lassen sich über direkte Verknüpfungen zu den entsprechenden Open-Source-Programmen bearbeiten. Bilder öffnet Scribus beispielsweise mit dem freien Photoshop-Pendant Gimp, für Texte greift das Programm auf den integrierten Texteditor oder alternativ auf den OpenOffice Writer zurück.

Fertige Layouts lassen sich in der Druckvorstufenüberprüfung auf Fehler prüfen und dann kann man die Farbauszüge erstellen. Die fertigen PDFs schreibt das Programm mit allen Extras, beispielsweise mit eingebetteten Schriftarten, einer automatischen Kompression oder mit integrierten Lesezeichen. Um EPS-Dateien zu importieren oder um PostScript-Dateien zu schreiben und zu drucken, wird GhostScript vorausgesetzt.

Inhaltlich überzeugt das Gratis-DTP-Programm. Wie die teueren Konkurrenten arbeitet Scribus schnell und gut mit wichtigen Quellformaten zusammen. Der Umgang mit den Elementen ist intuitiv in den jeweiligen Arbeitskontext eingebunden. Etwas schlechter schneidet Scribus dagegen beim Design des Programms ab. Ähnlich wie bei dem Grafikprogramm Gimp kann man die einzelnen Menüs und Unterfenster beliebig auf dem Desktop verteilen – für den Ein- bzw. Umsteiger recht gewöhnungsbedürftig. Weitere Informationen erhalten Sie auf folgenden Webseiten:

Scribus-Homepage: *http://www.scribus.org.uk/*
Deutsche Hilfe: *http://www.bomots.de/scribus*
Fehlerdatenbank: *http://bugs.scribus.net*

9.1.5 SciTE-Texteditor

Zum Programmieren, Schreiben von Shell-Scripten, bearbeiten von Konfigurationsdateien und für andere rein textorientierte Arbeiten ist natürlich der `vi` des UNIX-Freaks liebster Editor. Aber nicht jeder ist ein UNIX-Freak. Für diese Menschen gibt es den SciTE-Texteditor (den es sogar auch für Windows gibt). Er bietet alle Funktionen, die man von einem grafisch orientierten ASCII-Editor erwartet (er entspricht damit in etwa dem KEdit). Es lassen sich auch mehrere Dateien parallel öffnen und er besitzt sogar eine rudimentäre Projektverwaltung. Die Konfiguration erfolgt über einfache ASCII-Dateien. Es gibt eine Datei für globale Einstellungen, eine für benutzerspezifische und optional auch lokale Dateien in beliebigen Verzeichnissen.

Die Tastenbelegung orientiert sich am Windows-Standard, ist aber voll konfigurierbar. Die Tab-Taste fügt Leerzeichen anstelle von Tabs ein. Shortcuts wie Strg+L (Zeile ausschneiden), Strg+D (Zeile duplizieren) oder Strg+T (Zeilen tauschen) erleichtern das Arbeiten.

Speziell für Programmierer ist das so genannte Syntax Highlight gedacht. SciTE bietet Templates für Assembler, LaTeX, Java, C, JavaScript, Lisp, Perl, HTML, PHP, Python, Ruby, PovRay, Shell, TCL, SQL und noch mehr als 30 andere Sprachen. Außerdem können Sie sich Templates für eine andere Syntax selbst erstellen. Mit der Spracheinstellung für „Hypertext“ oder „PHP“ werden nicht nur die HTML-Befehle farblich markiert, sondern auch die PHP-Passagen durch einen leicht dunkleren Hintergrund markiert.

Die Such-Funktion erlaubt das Auffinden von Code-Schnipseln in der aktuellen Datei oder in allen Dateien eines angegebenen Ordners. Textstellen lassen sich mit und ohne reguläre Ausdrücke ersetzen (also auch da kein Bedarf für den `vi`) und das im gesamten Dokument oder nur in der definierten Selektion. Besonders nett: Bei gedrückter Alt-Taste erfolgt die Selektion im Spaltenmodus.

Eine weitere für Programmierer praktische Funktion finden Sie im *View*-Menü *Indentation Guides*. Damit werden Anfang und Ende von Abfragen oder Schleifen durch einen vertikalen Strich verbunden. Damit wird keine geschweifte Klammer mehr übersehen und der Code bleibt übersichtlich. Es reicht sogar, eine Klammer anzuklicken, schon wird die zugehörige andere Klammer gezeigt. Hilfreich ist auch das „code-folding“ von Funktionen, Abfragen, Schleifen oder Ähnlichem in Quelltexten. Dadurch kann man nicht benötigte Teile des Quelltextes verstecken und erhält mehr Übersicht bei langen Dateien.

Des Weiteren bietet SciTE noch Funktions-Tipps (Calltips), Abkürzungen für häufige Textpassagen, compilieren direkt aus dem Editor heraus mit Rückgabe der Meldungen, Unicode-Unterstützung (UTF-8) sowie Drucken mit Highlighting und Zeilennummern.

Der Link zur Projekthomepage lautet *http://www.scintilla.org/*.

9.1.6 GLabels: Aufkleber und Visitenkarten

Sicherlich lassen sich CD-Covers, Etiketten, Ordnerrücken und Visitenkarten unter Linux auch mit Office oder LaTeX gestalten, doch manchmal ist man einfach zu faul für so etwas. „GLabels" heißt die handliche Alternative, mit der das Gestalten von Etiketten wahre Freude macht. Das Programm ist sehr einfach und übersichtlich gestaltet. Eine Konfiguration ist nicht notwendig. Zuerst wählen Sie eines von über 100 verschiedenen Etikettenformaten aus. Die Auswahl reicht dabei von unterschiedlichen Briefaufklebern über CD-Labels und Ordnerrücken bis hin zu verschiedenen Visitenkarten. Wer eigene Etikettenformate erstellen will, dem hilft ein menügeführter Vorlagen-Designer beim Anlegen des neuen Layouts.

Ist das gewünschte Label-Format gewählt, kann das Label mit den in der Werkzeugleiste verfügbaren Elementen gefüllt werden. Neben geometrischen Objekten, Text und Bildern können auch Barcodes eingebunden werden, was besonders für den Einsatz als Firmenausweis interessant sein dürfte. GLabel generiert Barcodes automatisch aus einer vorher eingegebenen Zeichenfolge (28 verschiedene Typen von Kodes). Leider lassen sich keine Hintergrundfarben, -muster oder Bilder für das ganze Label festlegen. Wer also keinen weißen Hintergrund möchte, muss entsprechend farbiges Papier verwenden. GLabels finden Sie unter *http://glabels.sourceforge.net*.

9.1.7 Rechtschreibprüfung mit Ispell

Das Programm Ispell ist ein schnelles Kommandozeilen-Tool zur Rechtschreibprüfung. Gefundene Fehler werden angezeigt und mögliche Verbesserungen vorgeschlagen. Die Wörter des zu korrigierenden Textes werden dabei mit einem Wörterbuch Ihrer Wahl verglichen und Sie können interaktiv entscheiden, ob Sie den entdeckten Fehler korrigieren, das Wort dem Wörterbuch hinzufügen oder die Fehlermeldung ignorieren möchten.

Beim Aufruf von Ispell muss der Typ der Datei entweder als Option in der Kommandozeile oder als Endung des Dateinamens angegeben werden. Überprüfen eines deutschen Textes in ASCII-Schreibweise (die Umlaute sind aus zwei Zeichen zusammengesetzt):

```
$user: ispell -d deutsch -T ascii Datei
```

Überprüfen eines deutschen Textes im Latin1-Zeichensatz (die Umlaute werden durch 8-Bit-Zeichen dargestellt):

```
$user: ispell -d deutsch -T latin1 Datei
```

Überprüfen eines deutschen Textes im LaTeX-Format oder einer HTML-Datei:

```
$user: ispell -d deutsch -t Datei
$user: ispell -d deutsch -h Datei
```

Jedes Wort, das nicht im Wörterbuch steht, wird als Fehler interpretiert und in der obersten Zeile angezeigt. Stehen im Wörterbuch ähnlich geschriebene Wörter, werden diese als Verbesserung angeboten. Dabei erkennt Ispell vertauschte Buchstaben genauso wie Wörter vom gleichen Stamm. Man kann das Wort vollständig ersetzen oder eine der vorgeschlagenen Alternativen einsetzen. In einer Statuszeile am unteren Bildschirmrand Ihres Terminals finden Sie eine Reihe von Kommandos:

R (replace) ersetzt das fehlerhafte Wort im ganzen Text. Sie erhalten eine Eingabe-Aufforderung, an der Sie angeben können, wodurch das Wort ersetzt werden soll.

A (accept) akzeptiert das Wort für den Rest der Ispell-Session. Es erfolgt kein dauerhafter Eintrag in Ihr persönliches Wörterbuch.

I (insert) trägt das Wort in Ihr persönliches Wörterbuch ein.

U (uncapitalize) akzeptiert das Wort und fügt es in Kleinbuchstaben in das persönliche Wörterbuch ein.

0-n ersetzt das Wort durch Korrekturvorschlag 0, 1, . . . n.

L (lookup) schlägt ein Wort im Wörterbuch nach.

X speichert die Datei und beendet Ispell.

Q beendet Ispell, verwirft nach Bestätigung alle bisherigen Änderungen.

? ruft die Hilfe auf.

Praktisch ist auch, dass auf Wunsch direkt eine Sicherungskopie der überprüften Datei angefertigt wird. Mit ispell -b (backup) wird eine Kopie angelegt, die auf „.bak“ endet. Man braucht also nicht immer unbedingt ein Programm mit grafischer Oberfläche.

9.1.8 Tabellenkalkulation

Auch für die Kalkulation bietet Linux entsprechende Werkzeuge. Neben *3-2-1*, einem mit Lotus 1-2-3 kompatiblen Tabellenkalkulationsprogramm für UNIX (bis zu 100.000 Zeilen, 700 Spalten und 40 Funktionen) oder *kcalc* unter KDE hat sich *Gnumeric* etabliert, das aus dem Gnome-Projekt stammt, aber auch unter der KDE läuft. Es ging aus der GNU-Software *Oleo* hervor, einer Tabellenkalkulation, die sowohl im Texmodus als auch mit grafisch orientierter Oberfläche verfügbar ist und weiterhin für alle UNIX-Systeme gepflegt wird.

Das Ziel der Entwickler des Open-Source-Programms Gnumeric war es, möglichst viele nützliche Funktionen zu implementieren und so eine leistungsfähige und Ressourcen schonende Alternative zu kommerziellen Produkten zu bieten. Um den Umstieg zu erleichtern, wurde Gnumeric mit diversen

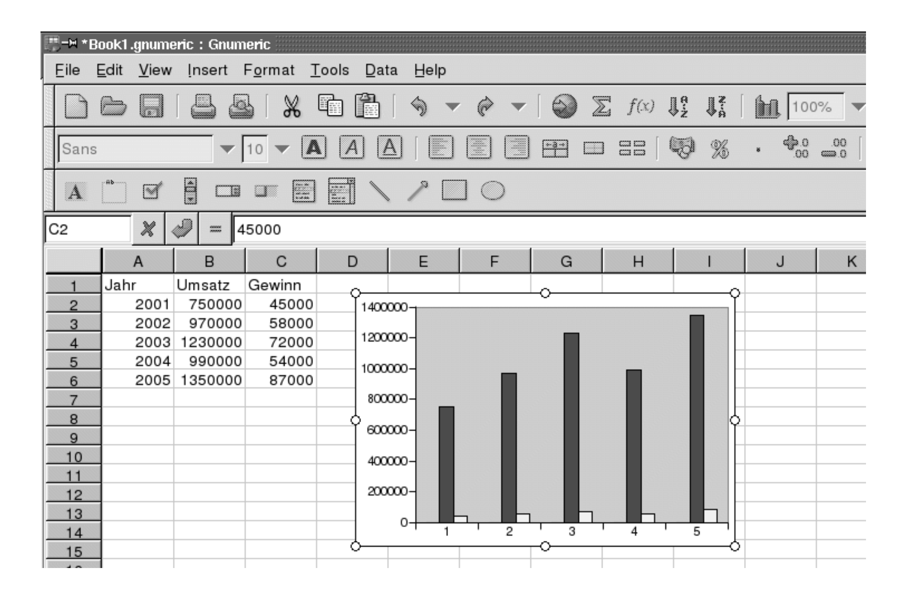


Abbildung 9.3: Spreadsheet mit Gnumeric

Im- und Export-Funktionen ausgestattet. Wenn Sie bereits mit einem Tabellenkalkulationsprogramm gearbeitet haben, wird Ihnen der Ein- oder Umstieg auf Gnumeric leichtfallen. Die zur Zeit der ManuScripterstellung aktuelle Version ist 1.6.2, die unter anderem Erweiterungen am Diagramm-Modul und an den Import- und Exportfähigkeiten enthält. Die Version basiert auf GTK 2.4 (eine Windows-Version soll bald kommen). Zurzeit wird auch an der Unterstützung von OpenDocument gearbeitet.

Gnumerics Im- und Export-Fähigkeiten unterstützen unter anderem Microsoft Excel, XML, HTML, Applix, Quattro Pro, PlanPerfect, Sylk, DIF, StarOffice und Lotus-123. Das eigene Format basiert auf XML, welches mittels gzip komprimiert wird.

Die Benutzeroberfäche und die Bedienung ähneln, wie schon angedeutet, stark dem Windows-Vorbild, wobei der Umfang der Funktionen dieses bereits weit überholt hat. Insbesondere die vielen Statistikfunktionen zusammen mit den vielen Chart-Varianten machen Gnumeric zu einem äußerst nützlichen Tool. Schön ist zum Beispiel, dass bei der Formeleingabe die entsprechenden Felder im Moment der Eingabe farbig umrahmt werden. Man sieht also gleich, ob man an die richtigen Felder referenziert.

Wie bei so manchen Linux-Programmen liegen die eigentlichen Qualitäten im Verborgenen. Nicht das Excel ähnliche Aussehen macht Gnumeric zur interessanten Alternative, sondern zum einen das zukunftsträchtige XML-Datenformat und für LaTeX-Anwender die einfache Exportmöglichkeit. So kann man seine Tabellen mit Gnumeric sehr einfach erstellen und dann nach LaTeX exportieren.

Aber auch HTML-Tabellen lassen sich erstellen – und zwar ohne den Overhead, der bei anderen Programmen erzeugt wird, sondern schon beinahe minimalistisches HTML.

Das Programm ist schnell, übersichtlich und sehr gut geeignet zur Einführung in die Funktionsweise eines Tabellenkalkulationsprogramms. Auch komplexe mathematische Aufgaben können damit gelöst werden; zu finden unter *http://www.gnome.org/projects/gnumeric/*.

9.2 Mind Mapper

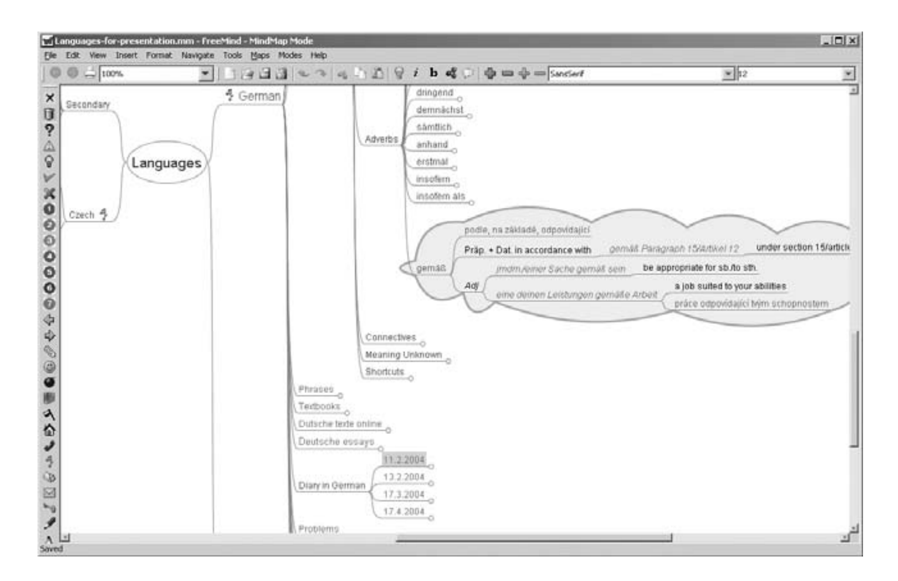


Abbildung 9.4: Den Mind Mapper freemind gibt es nicht nur für Linux.

Denken ist kein linearer Prozess, sondern verläuft vernetzt, sprunghaft und mit Assoziationen. Die Gedanken als Liste aufzuschreiben wird diesem Vorgang nicht gerecht. Mit „Mind Mapping“ können Sie Assoziationen grafisch darstellen. *Freemind* ist ein Linux-Mind-Mapper zum Katalogisieren von Wissen und Generieren von Ideen. Das Programm benötigt die Java-Laufzeitumgebung.

In Freemind können Sie Elemente (so genannte Knoten) verbinden, frei anordnen und mit Symbolen und Farben versehen. Auch Internet-Links lassen sich einfügen. Um bei grossen Mind Maps die Übersicht nicht zu verlieren, lassen sich einzelne Strukturäste durch Wolken bündeln.

Das Programm ist direkt und schnell zu bedienen, denn es konzentriert sich auf das Aufnehmen von Gedanken und Ideen in einem Wissensbaum, ohne mit überflüssigen Funktionen aufzuwarten. Die Darstellung ist sehr kompakt, so daß

vieles gleichzeitig sichtbar bleibt. Die Navigation kann komplett intuitiv über die Tastatur erfolgen (Mausbedienung geht auch). Die Pfeiltasten erlauben das Navigieren, Einfg fügt einen neuen Sibling-Knoten (untergeordnet) ein und um einen neuen Node auf derselben Ebene wie dem markierten Knoten einzufügen, genügt Enter.Zum Verschieben eines markierten Knotens in einem Ast dient Strg und die Pfeiltaste nach oben oder nach unten. Äste lassen sich mit der Leertaste ein- und ausklappen. Cut&Paste funktioniert auch so, wie man es erwartet. Fein.

Per Funtionstasten kann man Knoten einfärben und als „Ordner", „Frage" oder „Thema" kennzeichnen. Dazu gibt es noch ein paar hübsche bunte Icons und die Möglichkeit, Schriftfarben und -größen zu ändern. Mehrzeilige Nodes sind möglich, die Darstellung lässt sich zoomen. Nichthierarchische Querverbindungen von einem Node zum anderen lassen sich leider nicht ziehen und der erste Ast steht anfänglich fest in der Mitte.

Freemind speichert seine Daten in einem ganz gut lesbaren XML-Format und ist Open Source. Ist eine Mind Map fertig, exportieren Sie diese als HTML-, PDF- oder JPG-Datei. Download von *http://freemind/sourceforge.net*.

9.3 Projektplanung

Projektmanagement umfasst alle Methoden und Techniken, um Projekte standardisiert zu planen, zu steuern und letztlich erfolgreich zum Abschluss zu bringen. Unter Linux stehen mehrere Tools zur Verfügung, die zur Projektplanung und -kontrolle eingesetzt werden können.

Für den Projektmanager sind vor allem die Bereiche Ressourcenmanagement (die Zuordnung von Ressourcen, beispielsweise Mitarbeiter, Material oder Maschinen, nach Fähigkeiten und Verfügbarkeit auf die einzelnen Aufgaben), Kostenmanagement (Budgetüberwachung) und Zeitmanagement (Einhaltung des gesteckten Zeitrahmens) von Bedeutung. Er benötigt einfach bedienbare Tools, welche die Planung und Überwachung eines Projekts möglichst automatisieren, also Abhängigkeiten selbst auflösen und darstellen, Daten in verschiedenen Ansichten zeigen und vor allem warnen, wenn eine festgesetzte Größe Gefahr läuft, nicht mehr eingehalten zu werden.

Es gibt für Linux etliche Projektplanungsprogramme, darunter *Gantt-Project* zur Planung von Projekten mit Gantt-Charts, Export in HTML oder PDF sowie Unterstützung für 21 verschiedene Sprachen oder *Project/Translation*, ein Projektmanagement-System speziell für Übersetzungsbüros. Wir wollen nur zwei Tools herauspicken, die unserer Meinung nach den Bedarf gut abdecken.

9.3.1 Planner

Planner wird von seinem Autor mit Microsoft Project verglichen. Im Gegensatz zu diesem konzentriert sich Planner jedoch nur auf die Kernpunkte des Pro-

jektmanagements, andere Teile von Microsoft Project werden ausgeklammert. deshalb liegen die Schwerpunkte von Planner beim Zeit- und Ressourcenmanagement. Einzelne Aufgaben lassen sich erfassen, indem man Anfangs- und Endzeitpunkt sowie die Dauer der Aufgabe angibt. Planner hebt die Abhängigkeiten zwischen den einzelnen Aufgaben hervor und stellt auch den kritischen Pfad dar.

Mit Planner lassen sich einzelne Ressourcen definieren und Aufgaben zuordnen. Ein unabhängiger Ressourcenkalender zeigt freie Zeiten. Das erleichtert die Planung der Ressourcen innerhalb des eigenen Projekts. Der Punkt Kostenmanagement ist in Planner nicht enthalten, zudem beherrscht das Programm keine Reports in beliebiger Tiefe. Die Software ist übersichtlich und für Einsteiger einfach zu bedienen. Sie liefert eine Unterstützung für Planungsaufgaben, wenn der Projektleiter das Kostenmanagement anders durchführt – eignet sich also gerade für studentische Projekte. Weitere Informationen finden Sie auf der Homepage *http://www.simpleprojectmanagement.com/planner/*.

9.3.2 TaskJuggler

Das Projektplanungswerkzeug TaskJuggler ist an sich nur eine Sammlung von Bibliotheken und Kommandozeilen-Tools, es gibt aber auch ein GUI-Frontend. Für TaskJuggler wichtig sind die Bereiche Zeitmanagement, Kosten- und Ressourcenmanagement. Es plant selbstständig unabhängige Aufgaben und löst Konflikte auf. Beim Festlegen von Zeiten ist TaskJuggler flexibel und stellt auch arbeitsfreie Perioden im Gantt-Diagramm dar. Es unterstützt dabei sogar mehrere Zeitzonen. Für die Ressourcenplanung lassen sich Gruppen bilden und einzelnen Ressourcen Kosten zuordnen. Ebenso können Initial- und Endkosten für ein Projekt erfasst werden.

TaskJuggler bietet umfassende Reports mit vielfältigen Ordnungskriterien. Das Gantt-Diagramm zeigt die Zuordnung von Ressourcen zu einzelnen Tasks. TaskJuggler liefert eine Übersicht über die Personalplanung mit Verfügbarkeit, Auslastung und Kostenfaktor. Der Ressourcenkalender zeigt, welche Ressourcen zu welchem Zeitpunkt in welchem Umfang verfügbar sind.

TaskJuggler ist leistungsfähig genug für den professionellen Einsatz. Alle Komponenten des Zeit-, Ressourcen- und Kostenmanagements sind vorhanden, die Daten kann der Anwender trotzdem leicht eingeben und verwalten. Gelungen ist auch die Reporting-Funktion, die sehr übersichtlich den aktuellen Projektstatus mit den Parametern Zeit, Kosten und Ressourcen darstellt.

TaskJuggler ist Open Source (*http://www.taskjuggler.org/*).

9.4 Grafikprogramme

Werkzeuge zum Lesen von Bildern aus Digitalkameras, zum Scannen von Bildern, zum Erzeugen von Screenshots werden in Kapitel 7.5 ab Seite 305 vor-

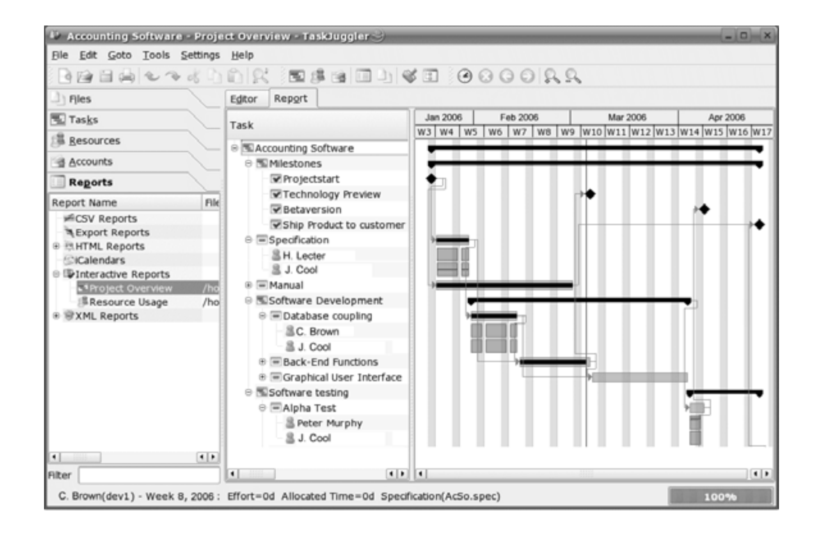


Abbildung 9.5: TaskJuggler zeigt, was er kann.

gestellt. Dort werden ab Seite 312 auch einige kleinere Tools zur Anzeige von Bildern sowie zur Durchführung einfacher Operationen vorgestellt. In diesem Abschnitt möchten wir Ihnen einige spezielle Programme vorstellen. Der Abschnitt über Gimp ist etwas länger, weil die meisten grafisch orientierten Programme ganz ähnlich bedient werden und wir bei anderen Programmen nicht mehr auf die Bedienung eingehen wollen.

9.4.1 Gimp, die Photoshop-Alternative

Hinter dem Kürzel Gimp (*GNU Image Manipulation Program*) verbirgt sich ein Programm zur Bildverarbeitung, das in vielerlei Hinsicht ähnliche und bisweilen sogar bessere Funktionen als kommerzielle Programme bietet. Gimp ist natürlich freie Software (GPL). Dieser Abschnitt gibt eine erste Einführung in Gimp 2.0. Auch wenn es aus Platzgründen nicht einmal ansatzweise möglich ist, die vielen Features von Gimp ausführlich zu beschreiben, so erhalten Sie doch Informationen zu elementaren Funktionen.

Wenn Sie keine Erfahrung mit Photoshop oder einem vergleichbaren Programm haben, können Sie sich vielleicht unter einem Bildverarbeitungsprogramm noch nicht allzu viel vorstellen. Mit Gimp können Sie Bitmap-Dateien in unterschiedlichen Formaten lesen und speichern, Fotos und eingescannte Bilder nachbearbeiten (Farb- und Helligkeitsverteilung ändern, aufhellen, den Kontrast verstärken etc.), diverse Filter anwenden, etwa um Kanten hervorzuheben oder optische Effekte zu erzielen, Teile eines Bilds markieren, ausschneiden, separat bearbeiten und zu neuen Bildern zusammensetzen (Fotomontage), neue Grafikelemente in ein Bild einarbeiten (Linien, Füllmuster, Farbverläufe, Texte etc.)

und die resultierenden Bilder drucken bzw. speichern. Gimp erlaubt es, Operationen zu automatisieren oder im Batch-Betrieb auf eine ganze Sammlung von Dateien anzuwenden.

Neben der Online-Hilfe gibt es im Internet unzählige weitere Informationsquellen und es sind auch eine Reihe von Büchern zu Gimp erhältlich. Das Erlernen der Funktionen von Gimp ist allerdings nur der erste Schritt zu tollen Ergebnissen. Zumindest ebenso wichtig ist das Verständnis, wie diese Funktionen in der Praxis sinnvoll angewandt werden können.

Gimp ist ein unglaublich vielseitiges und leistungsfähiges Programm (und hier kratzen wir nur an der Oberfläche). Dennoch ist die Gefahr groß, dass Sie das Programm nach einigen ersten Experimenten enttäuscht wieder von Ihrer Festplatte löschen. Das hat vor allem zwei Gründe: Gimp ist zwar ein mächtiges Werkzeug, Sie müssen sich aber intensiv einarbeiten, bis Sie es effizient nutzen können. Für gelegentliche Arbeiten ist das Programm nur bedingt geeignet. Zweitens stellt Gimp erhebliche Ansprüche an Ihre Hardware. Um komfortabel arbeiten zu können, sollten Sie über ausreichend Speicher, eine schnelle CPU und einen großen Monitor verfügen.

Auch unterscheidet sich die Bedienung von Gimp zum Teil erheblich von der Photoshops oder verwandter Produkte. Bei manchen Funktionen hat man den Eindruck, das Programm sei von Technikern für Techniker geschrieben. Zudem ist es für Photoshop-Umsteiger oft schwierig, die richtigen Funktionen im verschachtelten Menü zu finden.

Das Erstellen von (PostScript-)Dateien für professionelle Druckaufgaben ist mit Gimp schwieriger als mit anderen Programmen. Die Herstellung von Bildern mit Sonder- oder Schmuckfarben (Duplex-Bildformat) ist zwar prinzipiell möglich, die Funktionen sind aber wenig komfortabel. Eine direkte Unterstützung von Pantone-Farben fehlt ganz, soll aber eventuell in Zukunft in Form von kommerziellen Plug-Ins zur Verfügung gestellt werden.

- In Gimp gibt es zwei Hauptmenüs: ein kleines Menü im Gimp-Hauptfenster (Toolbox) und ein umfassendes Bearbeitungsmenü in jedem Bildfenster. Dieses Menü ist auch überall innerhalb des Bildfensters mit der rechten Maustaste zugänglich.
- Fast alle Funktionen von Gimp können per Tastatur ausgewählt und durchgeführt werden. Dabei tritt allerdings oft das Problem auf, dass sich der Tastaturfokus nicht im Bildfenster befindet, sondern im zuletzt benutzten Dialog (wo andere Tastenkürzel gelten).
- Wenn Sie im Gimp-Konfigurationsdialog *Oberfläche* die Option *Dynamische Tastenkürzel* aktivieren, können Sie die Tastenkürzel von Menükommandos ganz einfach ändern. (Die geänderten Tastenkürzel werden in der lokalen Datei `~/.gimp-n.n/menurc` gespeichert.)

- Wenn Sie eine schon begonnene Mausaktion abbrechen möchten, drücken Sie (bei noch gehaltener linker Taste) zusätzlich die rechte Taste. Lassen Sie dann zuerst die linke und dann die rechte Taste wieder los.

- Die exakte Mauspositionierung fällt in Gimp oft schwer. Verwenden Sie Hilfslinien! Der Mauszeiger rastet in unmittelbarer Nähe dieser Linien ein (wird quasi magnetisch angezogen). Zum Zeichnen von Hilfslinien klicken Sie mit der Maus eines der beiden Lineale an und ziehen es heraus. Zum Verschieben vorhandener Lineale aktivieren Sie zuerst den *Verschieben*-Button der Toolbox (Vierfachpfeil). Sie dürfen beliebig viele Lineale verwenden (Ausblenden mit *Ansicht* → *Magnetische Hilfslinien*).

- Mit einem einfachen Mausklick in die Toolbox wählen Sie die elementaren Gimp-Funktionen aus. Alle Toolbox-Werkzeuge können auch per Tastatur aktiviert werden. Zu vielen dieser Funktionen steht außerdem ein Optionendialog zur Verfügung, der normalerweise auch im Toolbox-Fenster angezeigt wird. Wenn das nicht der Fall ist, aktivieren Sie den Dialog durch einen Doppelklick auf den entsprechenden Button in der Toolbox.

- Darüber hinaus werden viele elementare Funktionen durch die Einstellungen der Dialoge *Pinsel* (Stift- und Pinselformen) und *Muster* (Füllmuster) beeinflusst. Diese Dialoge werden durch das Kontextmenükommando *Dialoge* geöffnet.

- Seit Gimp 2.0 können alle Dialoge (nicht aber die Bildfenster) per Drag&Drop ineinander verschoben und angedockt werden. Damit vermeiden Sie, dass Sie den Überblick über unzählige Gimp-Fenster verlieren.

- Wenn Sie ein Untermenü häufig benötigen, machen Sie daraus einfach einen eigenen Dialog. Dazu öffnen Sie das Menü im Bildfenster mit dem Kontextmenü. Nur dort (nicht aber im Hauptmenü) erscheint oberhalb eines jeden Menüs eine gestrichelte Linie. Sobald Sie die anklicken, wird das Menü zu einem eigenen Fenster.

- Alle Mal- und Zeichenkommandos berücksichtigen die in der Toolbox eingestellte Vorder- und Hintergrundfarbe. Der Dialog zum Verändern dieser Farben wird durch einen Doppelklick auf das jeweilige Farbfeld aufgerufen. Das kleine Symbol links unten stellt die Default-Farben Schwarz und Weiß wieder her; der Doppelpfeil vertauscht Vorder- und Hintergrundfarbe.

- Mit (Strg)+(Z) können Sie die letzten fünf Operationen rückgängig machen. (Strg)+(Y) stellt die zurückgenommenen Operationen wieder her. Im Konfigurationsdialog *Umgebung* können Sie die Anzahl der Undo-Ebenen einstellen.

- **Reglereinstellung:** Viele Effekte und Einstellungen werden durch Regler gesteuert. Diese Regler sind mit der Maus aber nur schwer exakt einzustellen. Verwenden Sie stattdessen die Cursor-Tasten!

- **Speichern:** Speichern Sie regelmäßig! Gerade während der ersten Experimente passiert es oft, dass Sie ungewollt Ihr Bild zerstören. Verwenden Sie bis zur endgültigen Version das Gimp-eigene `xcf`-Dateiformat. Dies ist das einzige Format, in dem wirklich alle Informationen gespeichert werden. Verwenden Sie Formate wie JPEG, GIF, TIFF, PNG etc. nur für das Endprodukt!
- In Gimp sind normalerweise zumindest ein halbes Dutzend Fenster geöffnet. Um das Chaos auf dem Bildschirm zu minimieren, sollten Sie für Gimp einen eigenen Desktop verwenden.

Kein Bildverarbeitungsprogramm kann ein unterbelichtetes oder verwackeltes Bild retten. Mit den Filtern von Gimp können Sie aber die Wahrnehmungsqualität eines Bilds spürbar verbessern. Neben den Funktionen zum Verändern der Bildgröße, zum Anpassen der Farben (Sättigung, Kontrast, Helligkeit, Farbton, Gammakorrektur usw.) besitzt das Programm zahllose Filter.

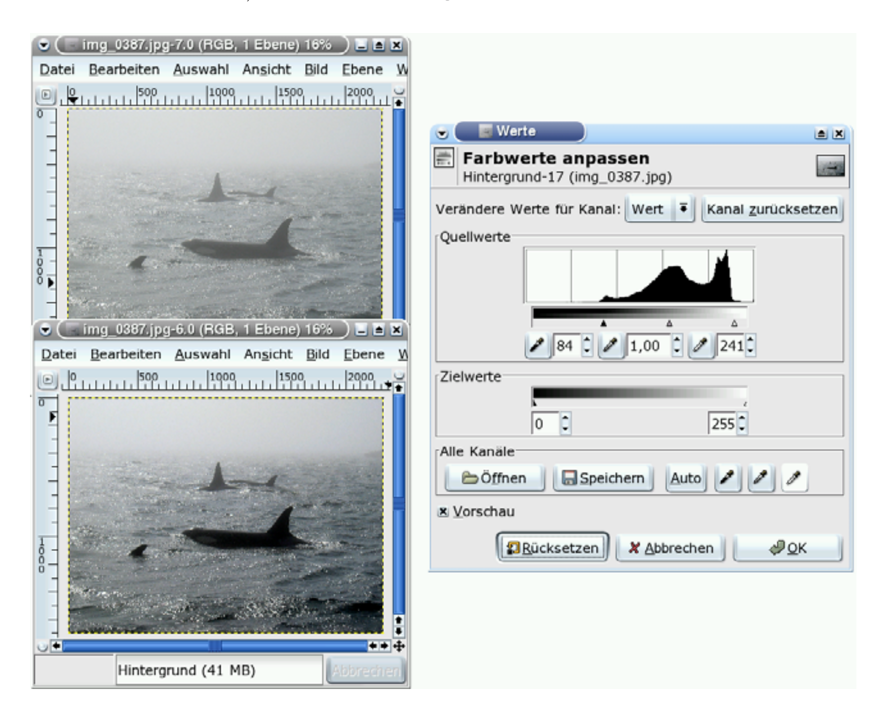


Abbildung 9.6: Bildbearbeitung mit Gimp: Weißabgleich

Wenn Sie Gimp nicht nur dazu verwenden möchten, um relativ einfache Effekte bei der Bearbeitung von Fotos zu erzielen, müssen Sie sich etwas intensiver mit seinen Grundfunktionen auseinandersetzen. Dieser und alle weiteren Abschnitte dieses Kapitels beschreiben die wichtigsten dieser Grundfunktionen.

Mit den Malwerkzeugen *Stift*, *Pinsel* und *Airbrush* können Sie sowohl ganz neue Grafiken erstellen als auch vorhandene Bilder verändern. Dabei sollte Ihnen aber immer bewusst sein, dass Gimp ein pixelorientiertes Bildverarbeitungsprogramm ist und kein vektororientiertes: Das Ergebnis einer Maloperation sind veränderte Pixel. Gimp speichert aber keinerlei Informationen über das gezeichnete Objekt. Um Text einzufügen, aktivieren Sie zuerst das *Text*-Tool und klicken dann mit der Maus an die ungefähre Position im Bild, wo der Text später erscheinen soll. Jetzt erscheint ein Dialog, in dem Sie die gewünschte Schriftart auswählen und den einzufügenden Text eingeben.

Fast alle Operationen lassen sich durch Optionen noch genauer steuern. Die wichtigsten Einstellungsmöglichkeiten finden Sie im jeweiligen Werkzeugdialog, der normalerweise unterhalb der Toolbox angezeigt wird. Wenn das nicht der Fall ist, hilft ein Doppelklick auf das Toolbox-Symbol weiter. Weitere Optionen befinden sich in den *Pinsel*- und *Muster*-Dialogen.

Die Anwendung von Malwerkzeugen auf das gesamte Bild ist die Ausnahme. Vielmehr sollen die meisten Operationen zumeist nur auf ganz bestimmte Teile des Bildes angewandt werden. Das Problem besteht darin, den gewünschten Bereich eines Bilds vorher möglichst exakt zu markieren. Der markierte Bereich wird durch einen blinkenden Rand (*marching ants*) gekennzeichnet. Gimp stellt eine ganze Reihe unterschiedlicher Methoden zur Auswahl, um damit Bereiche zu markieren.

Gimp-Bilder können aus mehreren Ebenen zusammengesetzt sein. Die einzelnen Ebenen können unterschiedlich groß und teilweise durchsichtig sein. Daraus ergeben sich eine Menge Berarbeitungsmöglichkeiten: Stellen Sie sich vor, Sie wollen per Fotomontage ein Aquarium nachbilden: Dazu verwenden Sie als Hintergrundebene ein Bild mit Wasser, in das Sie eventuell noch einige Luftblasen einfügen. Darüber ordnen Sie einige Ebenen mit Fischen an und ganz oben eine Ebene, die das Aquarium einrahmt. Im endgültigen Bild verdecken die Pixel der oberen Ebenen die Pixel der unteren Ebenen. Die Gesamtkomposition gelingt natürlich nur, wenn in jeder Ebene alle nicht relevanten Teile durchsichtig sind – und nicht etwa weiß!

Der große Vorteil von Ebenen besteht darin, dass jede Ebene individuell für sich verändert werden kann, ohne das restliche Bild zu beeinflussen. Generell gelten alle Gimp-Zeichenoperationen immer nur für die gerade aktive Ebene.

Auch nur einige Funktionen von Gimp zu erläutern – natürlich mit den dazu notwendigen Beispielbildern – würde dieses Buch auf doppelten Umfang anschwellen lassen. Deshalb müssen wir es bei diesem Appetithappen bewenden lassen.

9.4.2 Gnuplot

Seit vielen Jahren ist Gnuplot als eines der bekanntesten freien Plot-Programme verfügbar. Es handelt sich um ein plattformübergreifendes, kommando-

orientiertes, interaktives Plot-Programm, mit dem mathematische Funktionen und Datensätze in zwei- und dreidimensionalen Grafiken visualisiert werden können. Innerhalb kurzer Zeit gelangen Sie zu ansprechenden und druckreifen Grafiken. Die weitaus häufigere Anwendung ist die Visualisierung beliebiger Daten, sei es die Überwachung der Temperatur eines Ofens, die Auswertung von Logfiles oder die Darstellung von Börsenkursen. Mit Gnuplot steht dazu ein Werkzeug zur Verfügung, das auch in Kombination mit Shellscripten und anderen Tools ungeahnte Flexibilität entfaltet.

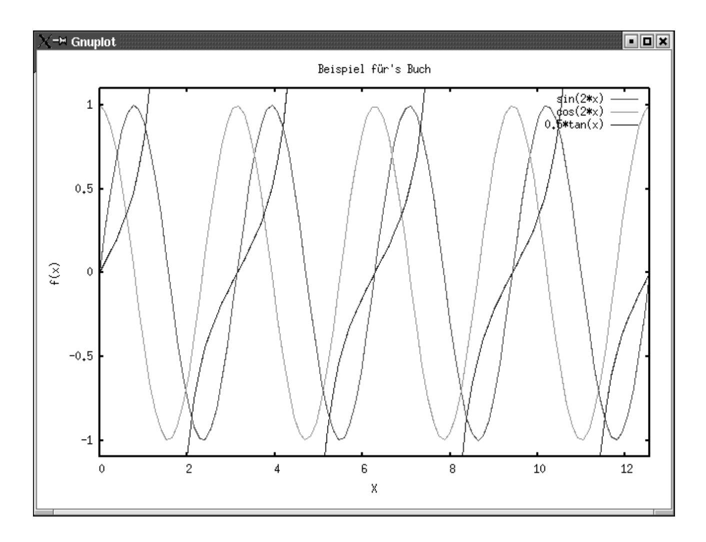


Abbildung 9.7: Gnuplot zeichnet trigonometrische Funktionen.

Für Bequeme gibt es zudem einige grafische Frontends. Dazu gehören Kile, ein Latex-Editor mit integriertem Gnuplot-Frontend, und Unignuplot. Hier soll aber die Kommandozeile im Vordergrund stehen. Gnuplot erlaubt dabei Visualisierungen im zwei- oder dreidimenisionalen Raum und geht auch bei extrem großen Datenmengen nicht in die Knie.

Für den Einstieg in Gnuplot gibt es mehrere Möglichkeiten. Zum einen existieren Tutorials und Handbücher, zum andereren ist das Ausprobieren und Hinzulernen der Bedienung bei Gnuplot eine spannende Sache. Zudem gibt es eine Demo-Sammlung, aus der Sie sich inspirieren lassen können. Probieren wir gleich einmal ein erstes Beispiel: Nach einer Begrüßung meldet sich Gnuplot mit seinem Prompt und es sind nun Eingaben möglich. Das Programm soll zunächst drei trigonometrische Funktionen ausgeben:

```
gnuplot> set xlabel "x"
gnuplot> set ylabel "f(x)"
gnuplot> set title "Beispiel für’s Buch"
gnuplot> plot [0:(4*pi)][-1.1:+1.1] sin(2*x), cos(2*x), 0.5*tan(x)
```

Nach Eingabe der letzten Zeile erscheint ein zweites Fenster mit der Grafikausgabe. Die Plot-Grenzen und Achsen-Skalierungen werden, sofern man nichts angibt, von Gnuplot über die autoscale-Funktion gewählt. Im Beispiel wurde der Plot-Bereich auf das x-Intervall (0, 4*pi) und das Funktionswerte-Intervall (-1.1, +1.1) eingegrenzt. Die Optionen mit dem Befehl *set* können immer wieder geändert oder erweitert werden. Soll danach die gleiche Grafik wieder dargestellt werden, eignet sich hierfür der Befehl *replot*. Natürlich sind unterschiedliche Linientypen möglich. Es gibt folgende *style*-Attribute des *plot*-Befehls: *lines*, *points*, *linespoints*, *dots*, *impulses*, *steps*, *fsteps*, *histeps*, *boxes*, *errorbars*, *xerrorbars*, *yerrorbars*, *xyerrorbars*, *boxerrorbars*, *boxxyerrorbars*, *vector*, *financebars*, *candlesticks*. Die Syntax lautet immer *plot „Datei“ with Style*. Häufig will man vorhandene Daten grafisch darstellen. Dafür gibt es in Gnuplot eine Schnittstelle, bei der aus ASCII-Dateien gelesen werden kann. Man gibt anstelle einer Funktion dann einfach die Datei an.

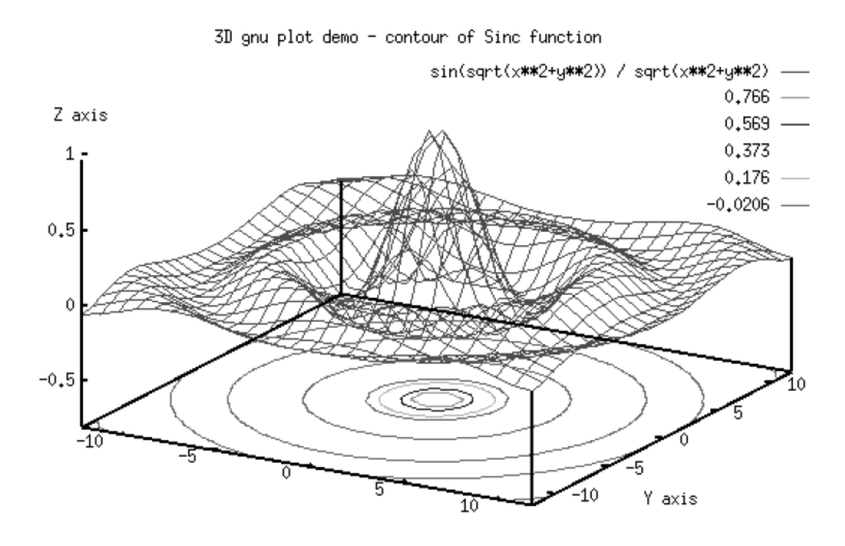


Abbildung 9.8: 3D-Grafik mit Gnuplot

Gnuplot beherrscht auch dreidimensionale Darstellungen (X-, Y- und Z-Achse). Für die dreidimensionale Darstellung gibt es in Gnuplot einige spezielle Befehle, die den fiktiven Blickwinkel des Benutzers auf die 3D-Grafik beeinflussen. Zu berücksichtigen sind dabei die möglichen Rotations-Intervalle. Die Anweisung *set hidden3d* blendet verdeckte Liniensegmente aus, um den 3D-Effekt noch zu verstärken. Soll Gnuplot unregelmäßig verteilte Datenpunkte darstellen, ist es empfehlenswert, zuerst die Lücken in der Datenpunktverteilung durch Interpolation zu schließen. Dazu dient der Befehl *set dgrid3d*. Ihm müssen die Anzahl der gewünschten Intervalle in x- und y-Richtung und ein Gewichtungsfaktor übergeben werden.

Um beim Beenden des Programms die Arbeit nicht zu verlieren, gibt es die Befehle *save* und *load*. Gnuplot speichert die benutzerdefinierten Funktionen, Variablen usw. in einer ASCII-Datei. Eine Funktion, die bei keinem Grafik-Programm fehlen sollte, ist die Druck-Funktion. Auch bei Gnuplot wird das Ausdrucken von Plots unterstützt. Dazu werden die Befehle *set output* und *set terminal* benötigt. Mit letzterem Befehl lässt sich bei Gnuplot die Grafikschnittstelle wählen (Default-Einstellung: X11). Andere Optionen sind unter anderem *epson, pcl5, postscript* und *latex*. In Verbindung mit *set output* kann somit eine Druckdatei erzeugt oder direkt gedruckt werden. Gnuplot ist zudem in der Lage, Plots als LaTeX-Kode zu exportieren.

Beim Start von Gnuplot sucht das Programm zunächst nach einer Initialisierungsdatei `.gnuplot` im Heimat-Verzeichnis. Dort können Gnuplot-Befehle untergebracht werden, die bei einem jeden Start von Gnuplot zuerst ausgeführt und da Gnuplot seine Befehle jederzeit auch aus einer Pipe oder einer Datei lesen kann, steht einer vollautomatischen Plotterei nichts im Wege.

Die aktuelle Version von Gnuplot sollte in jeder Linux-Distribution zu finden sein, muss also nur aus seinem Dornröschenschlaf geweckt werden. Weitere Informationen, das Handbuch und Versionen für alle gängigen Betriebssysteme sind unter folgenden Adressen auffindbar:

Homepage: *http://www.gnuplot.info*
Tutorial: *http://www.duke.edu/hpgavin/gnuplot.html*
Kurs: *http://userpage.fu-berlin.de/~voelker/gnuplotkurs/gnuplotkurs.html*
Manual: *http://www.tu-chemnitz.de/urz/anwendungen/grafik/gnuplotdoc.html*
FAQ: *http://www.ucc.ie/gnuplot/gnuplot-faq.html*

9.4.3 LabPlot

LabPlot geht in seiner Leistung etwas über Gnuplot hinaus, ist dafür aber auch komplexer in der Bedienung. Deshalb hat es eine komfortable grafische Oberfläche. Im Übrigen dient Labplot wie Gnuplot zur Darstellung und Auswertung zweidimensionaler und dreidimensionaler Funktionen und Daten. LabPlot erlaubt ihnen mit mehreren Plots zu arbeiten, von denen jeder mehrere Graphen besitzen kann. Die Graphen können aus Daten oder aus Funktionen generiert werden. Das Programm kann flexibel Daten in verschiedenen Formaten lesen und schreiben.

Alle Einstellungen eines gesamten Sets von Plots lassen sich in Projektdateien speichern. Diese Projektdateien werden über die Kommandozeile, über das Dateimenü oder per Drag&Drop geladen. Jedes Objekt (Titel, Legende, Achsen, Achsenbeschriftung) kann mit der Maus gezogen werden. Es werden alle Funktionen und Konstanten der GNU Scientific Library (GSL) unterstützt. Mit LabPlot lassen sich Oberflächen zeichnen (mit „hidden lines“) sowie Polar Plots,

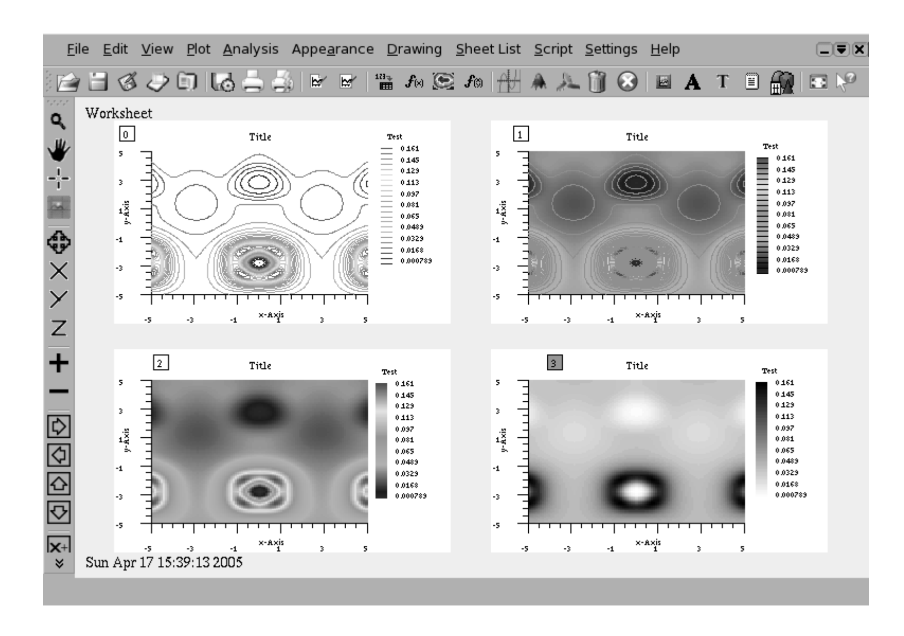


Abbildung 9.9: Ein Feature von Labplot

Ternary Plots und Tortendiagramme aus Funktionen und Datendateien erzeugen.

Die Einstellungen des Plots oder Graphen können leicht per Menü geändert werden und es lassen sich jederzeit zusätzliche Datensätze und Funktionen (Graphen) einfügen, die im gleichen oder einem anderen Plot dargestellt werden können. Detaillierte Dialoge für alle Einstellungen werden per Doppelklick geöffnet und jedes Objekt läßt sich mit der Maus platzieren. Weiterhin beherrscht Labplot Operationen zur Analyse von Daten und Funktionen sowie zur Mittelung, Glättung und Kürzung von Daten. Weitere Möglichkeiten sind die Komprimierung und die Analyse von Daten (Periodizitäten, Spitzen, Interpolation, Integration, Regression bis zur zehnten Ordnung, nichtlineare Anpassung, Fourier-, Hankel- und Wavelet-Transformation sowie Faltung.

Die Arbeitsblätter lassen sich als Bilder (PS, EPS, SVG, PDF) exportieren oder in andere Formate konvertieren (über pstoedit or ImageMagick). Auch der Import bzw. Export von Daten in und von Datenbanken ist möglich. Die Homepage von Labplot finden Sie unter *http://labplot.sourceforge.net/index-de.html*.

9.4.4 XFig, der Vektorzeichen-Klassiker

Bildbearbeitungsprogramme wie GIMP stoßen rasch an ihre Grenzen, wenn es darum geht, Diagramme und technische Zeichnungen zu erstellen. Vor allem zei-

gen sich Probleme beim Skalieren – je größer man die Zeichnung macht, desto „pixeliger“ wird sie. Der Vorteil von Vektor- gegenüber Bitmap-Grafiken liegt darin, dass sie nicht einzelne Bildpunkte, sondern mathematische Beschreibungen der Formen speichern. Dadurch lassen sich Objekte unbegrenzt skalieren, Eigenschaften von Objekten bleiben erhalten und können nachträglich modifiziert werden.

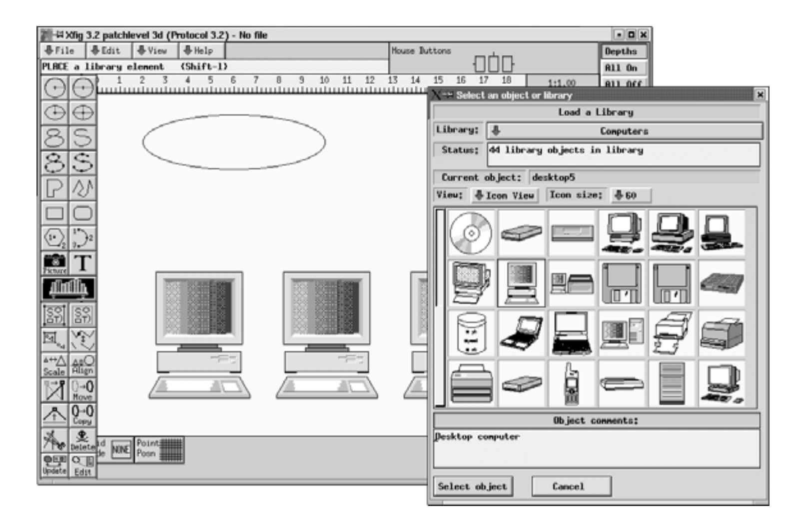


Abbildung 9.10: Bibliotheksobjekte auswählen mit Xfig

Älter als Linux selbst und ein echter Klassiker in puncto Vektorgrafik ist Xfig. Alle wichtigen Zeichenelemente befinden sich am linken Rand: Kreise, Linien, Vielecke und auch ein Button zum Einfügen von Bildern sind vorhanden. Xfig besitzt auch, wie beispielsweise Visio, eine Fülle verschiedener Bibliotheken für diverse Diagrammtypen. außerdem können Sie auch selbst entworfene Bilder in verschiedenen Pixelformaten in die Grafik importieren.

Um ein neues Objekt zu zeichnen, wählen Sie zunächst den Objekttyp, klicken dann mit der linken Maustaste ins Fenster und ziehen die Größe zurecht. Bei Kreisen, Ellipsen und Rechtecken lassen Sie die Maustaste einfach los, um das Objekt zu verankern. Bei komplexeren Figuren, wie z. B. Polygonen, setzen Sie über Klicks mit der linken Maustaste einzelne Punkte und klicken beim letzten Punkt mit der mittleren Maustaste zum Beenden.

Xfig verfügt über Funktionen, um Objekte zu gruppieren – entweder einen Bereich oder selektiv einzelne Elemente. Nur das Aufrechterhalten der Verbindungen zwischen einzelnen Objekten ist etwas mühsam. Für alle Funktionen zum Verschieben, Verändern, Löschen oder Skalieren von Objekten müssen Sie zunächst das entsprechende Icon in der Werkzeugleiste wählen und dann genau einen der Ankerpunkte des Objekts mit der Maus treffen. Andererseits genügt ein Klick auf den *Edit*-Button und schon öffnet sich ein Menüfenster,

das Ihnen auf einen Blick nahezu alle Attribute des Objekts zur Bearbeitung anbietet. Die fertige Grafik wird normalerweise in Xfigs eigenem Format gespeichert, aber auch der Export in gängigen Grafikformaten ist möglich. Über das Druckmenü wird die Orientierung der Grafik und die Größe festgelegt.

Das Zeichnen selbst geht mit Xfig dank der relativ flachen Menü-Hierarchie leicht und schnell von der Hand. Die Bibliotheken für einzelne Diagrammtypen machen teilweise einen etwas verspielten Eindruck. Sein Alter merkt man Xfig weniger am Fehlen wichtiger Funktionen an, eher am ziemlich altmodischen Erscheinungsbild der grafischen Oberfläche. Wer aber nur alle Jubeljahre mal ein Diagramm braucht, kommt mit Xfig schnell zu einem brauchbaren Ergebnis (*http://www.xfig.org*).

9.4.5 Vektorzeichenprogramm Dia

Dia ist ein Vektorzeichenprogramm zum Erstellen strukturierter Diagramme; seine Funktionalität ist mit der proprietären Software „Visio“ vergleichbar. Es dient wie Xfig dazu, strukturierte Diagramme aufzubauen, die aus Standardobjekten zusammengesetzt sind. So gibt es unter anderem Modi für Flowcharts, Netzwerk- und UML-Diagramme. Es gibt jedoch auch einige generische Funktionen, mit denen einfache Zeichnungen erstellt werden können. Das Programm verfügt über eine solide Zahl an Ausgabeformaten. Da es auf dem GTK+-Toolkit aufbaut, ist es auch unter Windows verfügbar. Sehr positiv fällt die Erweiterbarkeit auf. Sollte ein bestimmter Diagrammtyp noch nicht unterstützt werden, so können neue Objekttypen über XML-Dateien hinzugefügt werden.

Dia präsentiert sich als typisches GTK-Programm. Auf den ersten Blick glaubt man, aus Versehen Gimp gestartet zu haben. Das Werkzeugpanel sieht, von einigen Sonderfunktionen abgesehen, fast identisch aus. Die obere Hälfte bietet eine Auswahl der grundlegenden Zeichenfunktionen (Linien, Kurven, Rechtecke etc.), Text- und Bildeinbettung sowie eine Verschiebe- und eine Zoomfunktion. Darunter ist ein Optionsmenü angesiedelt, das eine Reihe von Bibliotheken für spezielle Diagrammtypen anbietet. Wie bei älteren Gimp-Versionen erreichen Sie die meisten Funktionen über das Kontextmenü. Es erscheint, wenn Sie mit der rechten Maustaste in das Diagramm klicken. Wem das zu umständlich ist, kann dieses und viele andere Menüs auch abreißen und als separate Fenster auf dem Desktop platzieren.

Wie bei Visio werden die einzelnen Objekte durch „Gummifäden“ verbunden, von denen es etliche Typen gibt (gerade, gebogen, eckig, polygon etc.). Man wählt dazu eine Verbindungslinie aus und „klebt“ sie mit je einem Ende an beide zu verbindenden Objekte. Danach dehnt sich die Line beim Verschieben der Objekte passend aus – sofern sie richtig verankert wurde. Über die grünen Ankerpunkte skalieren Sie die einzelnen Objekte; wenn Sie ein Symbol hingegen verschieben wollen, klicken Sie mit der Maus in den Bereich zwischen den grünen Punkten und ziehen es an die gewünschte neue Stelle. Weitere Eigenschaften legen Sie fest, indem Sie auf ein Objekt einfach doppelklicken. Abhängig vom

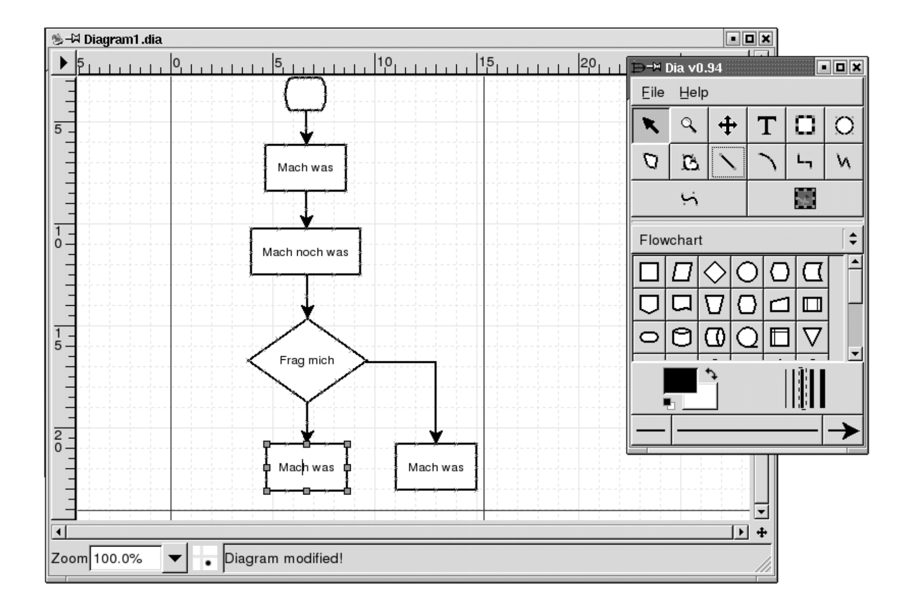


Abbildung 9.11: Dia wirkt wie eine Mischung von Gimp und Xfig.

Objekttyp konfigurieren Sie im erscheinenden Dialog weitere Dinge wie Linienbreite und -farbe, Füllfarbe oder Transparenz. Das Kopieren und Duplizieren von Objekten funktioniert wie bei einem Texteditor mit Strg-C und Strg-V.

Die Icon-Bibliotheken fallen im Vergleich zur Fülle von Xfig etwas spartanisch aus. Aber diese Schwäche gleicht Dia durch eine große Stärke wieder aus: Es bietet die Möglichkeit, über XML und SVG benutzerdefinierte Diagramm-Bibliotheken in das Standardmenü zu integrieren. Dia bietet das Arbeiten mit Ebenen an – so können Sie Objekte getrennt bearbeiten und auch übereinander einblenden.

Das fertige Diagramm speichern Sie im programmeigenen Format (.dia), einem komprimierten XML, oder Sie exportieren es in andere Pixelgrafik-Formate.

Dia punktet vor allem auf dem Gebiet der praktischen kleinen Einfälle, die das tägliche Zeichnerleben einfacher machen. So ist die Funktion, mit der einzelne Objekte fest miteinander verbunden werden können, schlichtweg einfach und genial. Soll eine Grafik häufiger geändert werden, macht diese Funktion einem das Leben um einiges leichter, da das Grundgerüst der Zeichnung sehr flexibel ist. Die Grafik kann umgestaltet werden, ohne dass man umständlich alle Verbindungen lösen, die Objekte bewegen und wieder verbinden müsste. Die Homepage von Dia finden Sie unter *http://www.gnome.org/projects/dia/*.

Das Programm Kivio ist als Teil von KOffice nur unter Linux mit KDE lauffähig und bietet eine ähnliche Leistung.

9.5 Symbolische Mathematik

9.5.1 Euler

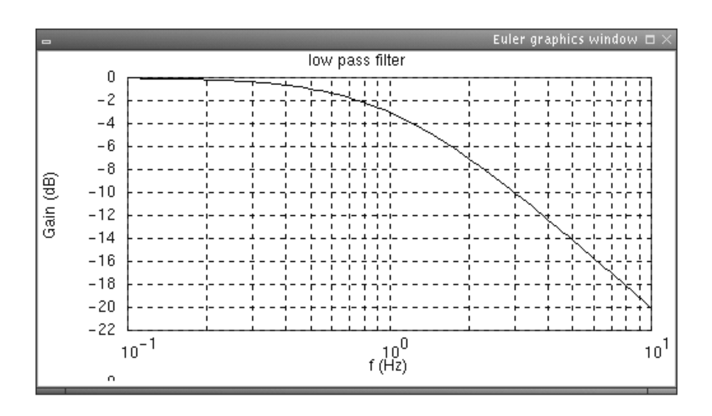


Abbildung 9.12: Kurven zeichen mit Euler

Euler ist ein Programm zum schnellen und interaktiven computergestützten Rechnen mit reellen und komplexen Zahlen sowie Matrizen. Es kann zwei- und dreidimensionale Funktionen zeichnen und sogar animieren. Die Benutzeroberfläche, die eingebaute Hilfe und die Programmiersprache von Euler arbeiten derzeit noch in Englisch. Auf Deutsch steht das Euler-Handbuch von H. D. Gellißen zur Verfügung. Das Programm steht unter der GPL. Euler existiert in Programmversionen für Windows und Linux.

Euler ist ein *numerisches* Laboratorium, weil Euler keine symbolischen Transformationen vornimmt wie z. B. Maple. Die Stärken des Programms liegen daher in der viel höheren Geschwindigkeit und in der einfachen Handhabung großer Datenmengen. Am stärksten verwandt ist Euler mit *MatLab* und ähnlichen Tools. Die Entwicklung begann, als auch *MatLab* noch in den Kinderschuhen steckte. Daher hat Euler zwar den gleichen Ansatz, ist jedoch nicht voll kompatibel. Einige Features von Euler sind:

- reelle und komplexe Skalare und Matrizen (auch Intervalle),
- eine Programmiersprache mit lokalen Variablen, Standardwerten für Parameter, Funktionen mit variabler Anzahl von Parametern,
- zwei- und dreidimensionale Graphen,
- beschriftete Plots aller Art,
- Animationen,
- numerische Integration und Differenziation,

- statistische Funktionen und Tests,
- Differenzialgleichungen,
- Funktions-Minimierer (Brent, Nelder-Mean),
- Simplex-Algorithmus,
- Interpolation und Approximation,
- Lösen der Wurzeln von Polynomen,
- schnelle Fourier-Transformation (FFT),
- exaktes Skalarprodukt durch Benutzung eines langen Akkumulators,
- Export von Grafiken nach PostScript.

Was kann man mit Euler anfangen? Nehmen Sie an, Sie hätten eine nichttriviale Funktion und möchten mit dieser eine Kurvendiskussion durchführen. Mit einem der Plot-Kommandos von Euler können Sie den Graph der Funktion zeichnen lassen. Mit weiteren Tools bestimmen Sie Nullstellen oder lokale Extrema. Sie könnten die Funktion auch integrieren oder ganze Kurvenscharen (Funktion mit verschiedenen Parametern) als einen Satz von Graphen oder als dreidimensionale Graphen erzeugen.

Ein anderes Beispiel: Angenommen, Sie haben eine Datei mit Messwerten. Euler kann diese Daten aus der Datei lesen und zeichnet daraus Plots, erzeugt Polynome, führt weitere Berechnungen aus usw. Oder Sie wollen einen numerischen Algorithmus überprüfen. Sie schreiben einen Prototypen dieses Algorithmus in der Euler-Programmiersprache (Dies ist gewöhnlich schneller erledigt als bei Benutzung einer klassischen Programmiersprache.) und testen diesen aus. Das funktioniert interaktiv und Sie können auch Graphen benutzen, um den Algorithmus zu visualisieren.

Euler ist ein ideales Werkzeug für Aufgaben wie:

- Untersuchung und Diskussion von Funktionen mit einer realen oder komplexen Variablen,
- Veranschaulichung von Flächen durch Parametrisierung,
- lineare Algebra und Eigenwert-Berechnung,
- Überprüfung numerischer Algorithmen,
- numerische Lösung von Differentialgleichungen,
- Berechnung von Polynomen,
- Studium der Intervall-Arithmetik,

- Generieren und Überprüfen von Sound-Dateien.

Aber Sie müssen für die Anwendung eine neue Sprache lernen. Es ist keine schwierige Sprache, aber es ist trotzdem ein wenig Aufwand damit verbunden. Im Prinzip ist Euler eine Matrixsprache. Es existieren sehr mächtige Funktionen, um Matrizen und Vektoren zu verwalten und zu modifizieren. Natürlich kann man das für lineare Algebra und Geometrie benutzen. Der Haupteinsatz liegt jedoch in der Erstellung von Daten- und Funktionstabellen.

Nach dem Start von Euler können Sie in der Befehlszeile Kommandos eingeben, die Euler berechnen oder ausführen soll. Die Zeile dient außerdem zur Eingabe von Funktionen, sofern diese nicht aus externen Dateien gelesen werden. Probieren wir etwas ganz Einfaches:

```
>(5*4+3/2)/4^2
      1.34375
```

Euler berechnet das Ergebnis sofort. Natürlich werden die üblichen Rechenregeln beachtet. Beispielsweise wird `4^2` (steht für 4 hoch 2) vor der Division berechnet. Euler kennt auch die üblichen mathematischen Funktionen. Außerdem kennt es komplexe Zahlen und verschiedene Ausgabeformate (short, long, longest, fract); z. B.:

```
>longestformat; 1/3
            0.3333333333333331
>fracformat; 1+1/2+1/3+1/4+1/5
             137/60
```

Intern wird jedoch immer mit `double`-Zahlen gerechnet. Auch Vektoren lassen sich recht einfach erzeugen, beispielsweise eine Reihe von Zufallszahlen mittels der Anweisung „`z=random(1,100);`“. Falls Sie das Semikolon vergessen haben, werden die hundert Zahlen ausgegeben. Nun haben wir einen Vektor z mit gleichverteilten Zufallszahlen zwischen 0 und 1 erzeugt. Man kann nun den Mittelwert und die Streuung berechnen:

```
>shortformat; m=sum(z)/100
     0.509271
>sqrt(sum((z-m)^2)/99)
     0.285003
```

Derselbe Effekt lässt sich auch mit den eingebauten Funktionen `mean(z)` und `dev(z)` erreichen. Da Funktionen auf ein jedes Element eines Vektors wirken, können Sie einen einfachen 2D-Plot folgendermaßen erzeugen:

```
>t=-3:0.01:3; xplot(t,sin(2*t)*t);
```

t ist ein Vektor der Werte -3.00, -2.99, ... 2.99, 3.00. Die Funktion *sin(2*t)*t* wird dann auf diese Werte angewendet und das Ergebnis als Kurve gezeichnet.

`xplot` erwartet zwei Vektoren mit x- und y-Koordinaten der Punkte. Man kann auch parametrische Kurven plotten, z. B.:

```
>hold; plot(sin(4*t),cos(5*t)); hold;
```

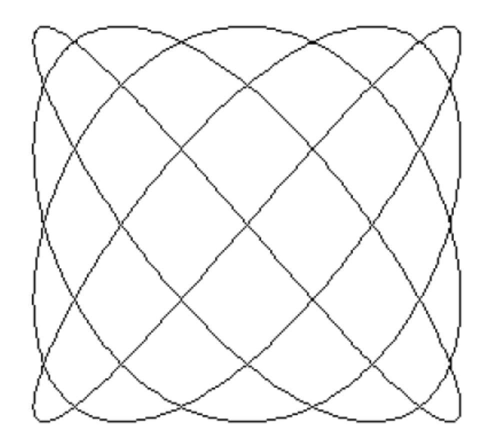


Abbildung 9.13: Graph für plot(sin(4*t),cos(5*t))

Das Ergebnis sehen Sie im Bild. Der Befehl `hold` sorgt dafür, dass kein Rahmen gezeichnet wird. Selbst definierte Funktionen lassen sich direkt in der Kommandozeile eingeben oder von externen Dateien laden. En Beispiel für eine interaktive Definition:

```
>function f(x)
return x^2 - 3x + 5;
$endfunction
```

Durch die Änderung des Cursors wird signalisiert, dass sich Euler im Eingabemodus befindet. Die Variable x hat keinen Datentyp. Die Funktion ist daher auf alle Datentypen anwendbar, die Euler verwenden kann.

Euler kennt zahllose eingebaute Funktionen, so auch für die bereits erwähnten Möglichkeiten der Bestimmung von Nullstellen, Minima und Maxima. Auch stehen nahezu alle bekannten numerischen Verfahren zur Verfügung. Weitere Informationen finden Sie unter:
http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/euler/german/Start.html
http://computing.ee.ethz.ch/sepp/euler-1.60.6-mo/german/

9.5.2 MuPAD

MuPAD nennt sich „mathematisches Expertensystem“. Es lassen sich damit symbolisch-algebraische Berechnungen (Formelmanipulationen, Rechnungen mit exakten Ergebnissen), aber auch numerische Berechnungen mit beliebiger Genauigkeit durchführen. Zusätzlich zu einer breiten Palette vordefinierter Funktionen aus nahezu allen mathematischen Fachgebieten und der Möglichkeit, das System mit eigenen Programmen und Datentypen zu erweitern, bietet MuPAD zwei- und dreidimensionale Visualisierungen für statische Objekte und zeitliche Abläufe (Animationen).

Auf Linux-, Windows- und Macintosh-Systemen bietet MuPAD dazu ein flexibles Arbeitsblatt-Konzept zum Erstellen mathematischer Dokumente, die Texte, Grafiken, Formeln, Berechnungen und mathematische Visualisierungen und Animationen vereinen. Dieses Computer-Algebra-System wurde ursprünglich von der MuPAD-Forschungsgruppe der Universität Paderborn unter der Leitung von Prof. Benno Fuchssteiner entwickelt. Im Jahr 1997 wurde die Entwicklung von der Firma Sciface Software übernommen, einer Ausgründung aus der MuPAD-Forschungsgruppe.

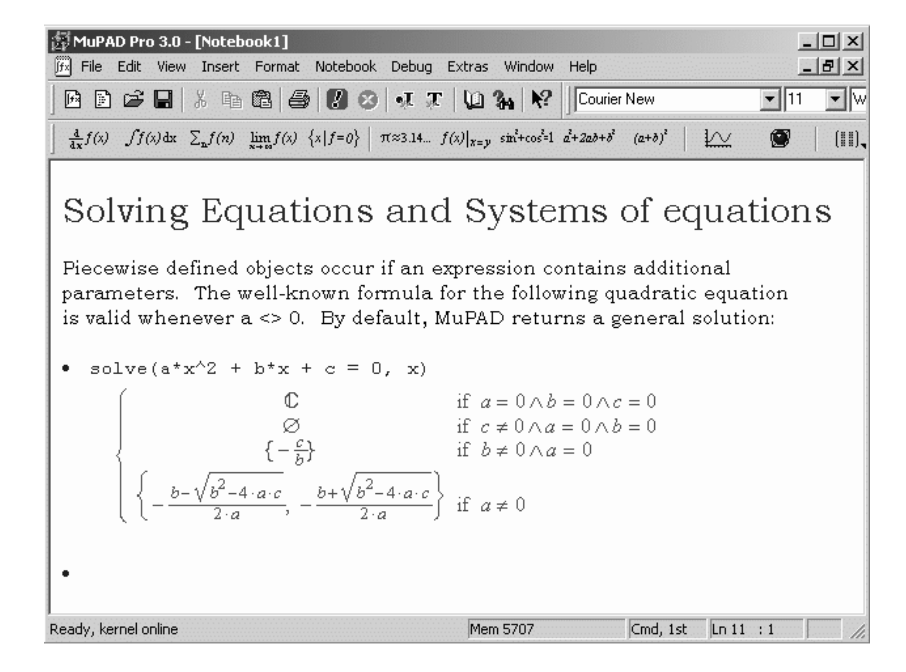


Abbildung 9.14: Die MuPAD-Oberfläche

MuPAD Pro 3.2 für Linux gibt es seit Ende 2005. Es verfügt über ein neues XML-basiertes Arbeitsblatt-Konzept. Um MuPAD Pro 3.2 zu benutzen, brau-

chen Sie einen Lizenzschlüssel. Eine Testlizenz bekommen Sie unter *http://edu.mupad.de/download/index_trial.php?request=trial&lang=de*.

Der zentrale Bestandteil des MuPAD-Systems ist der „Kern“. Der Kern stellt u. a. die folgenden Aspekte zur Verfügung:

- Die Arithmetik erlaubt beliebig große ganze oder gebrochene Zahlen oder Gleitkommazahlen beliebiger Genauigkeit.
- Der Parser nimmt Anwendereingaben entgegen und bereitet sie für die Auswertung vor.
- Der Evaluierer wertet Benutzereingaben aus, arbeitet Programme ab und führt grundlegende Vereinfachungen durch.

Der MuPAD-Kern stellt eine objektorientierte Programmiersprache mit Pascal-ähnlicher Syntax zur Verfügung. Nahezu die gesamte mathematische Expertise des Systems ist in dieser Sprache implementiert und daher unabhängig vom Betriebssystem. Die meisten dieser Funktionen sind in Bibliotheken organisiert. Ergebnisse lassen sich in RTF oder HTML exportieren.

MuPAD bietet eine extensive Grafik-Bibliothek, die auf eigenen Datentypen für grafische Objekte wie Punkten oder Funktionsgraphen aufbaut. Die entstehenden Grafiken lassen sich interaktiv manipulieren – beispielsweise können Sie den Beobachtungspunkt verschieben, die Farben neu setzen oder sogar grundlegende Definitionen wie die geplottete Funktion ändern. Grafiken lassen sich in diversen Formaten speichern oder auch drucken.

MuPAD lässt sich nicht nur durch neue Funktionen und Datentypen auf Bibliotheksebene erweitern, sondern auch der Systemkern kann durch eigene C++-Programme erweitert werden, die „dynamische Module“ genannt werden. Aus der Sicht des Anwenders gibt es kaum Unterschiede zwischen einem dynamischen Modul und einer Bibliothek. Ein Modul enthält Modulfunktionen, die mit Hilfe des MuPAD API (MAPI) in C++ geschrieben sind. Darüber hinaus können Module auch „normale“ Bibliotheksfunktionen enthalten.

Weitere Informatioen finden Sie unter *http://www.mupad.de/*.

9.6 Chemie und Biologie

9.6.1 Periodensystem der Elemente

Zum Periodensystem der Elemente gibt es einige Programme. Wir wollen Ihnen hier drei davon vorstellen.

GPeriodic ist ein freies, quelloffenes Periodensystem der Elemente. Für jedes angeklickte Element wird eine Reihe von Eigenschaften angezeigt. Derzeit sind 118 Elemente gelistet. Homepage: *http://gperiodic.seul.org/*.

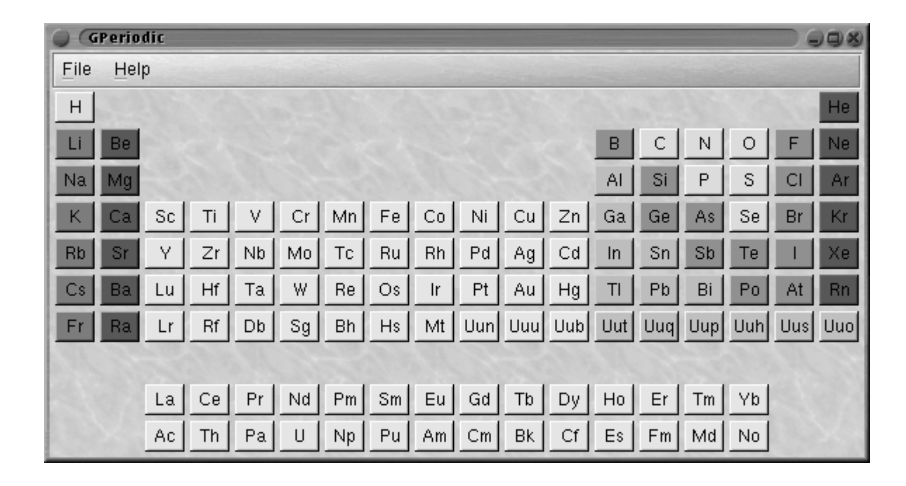


Abbildung 9.15: Das Einstiegsbild von GPeriodic

Elem ist ein ähnliches Programm. Durch einen Klick auf ein Element erscheint ein Dialogfenster mit weiteren Informationen zum entsprechenden Element. Weiterhin kann die Summe der Atommassen aller angeklickten Elemente gebildet werden. Damit lässt sich die Molarmasse von Molekülen berechnen. Download unter *http://elem.sourceforge.net/index-de.html*.

Am ausgereiftesten ist wohl **Kalzium** aus dem KDE-Edutainment-Projekt. Einfach und übersichtlich sind die derzeit bekannten chemischen Elemente aufgelistet. Auch hier erscheinen per Mausklick weitere Informationen zum ausgewählten Element. Dafür wird ein neues Fenster geöffnet, welches dreigeteilt Informationen allgemeiner Art (wie Name, Symbol, Atommasse oder Dichte), über den Aggregatszustand (Schmelz- und Siedepunkt) und zu Energien (Ionisationsenergie und Elektronegativität) anzeigt. Dort gibt es je nach Element auch einen mit „Internetsuche" beschrifteten Knopf. Über diesen gelangt man auf die Seiten mit weiteren Infos zum entsprechenden chemischen Element.

Im Einstellungsdialog kann man auch die Farben ändern, um Gruppen nach Merkmalen wie Säureverhalten oder Aggregatszustand darzustellen. Kalzium besitzt zusätzlich eine Zeitleiste. Nach Anklicken dieses Knopfes erscheint unterhalb des Periodensystems ein Schieberegler, welcher auf dem aktuellen Jahr steht. Nun kann man in der Geschichte zurückgehen und sehen, welche Elemente zu welcher Zeit schon bekannt waren. Weiterhin gibt es ein kleines Quiz und die Möglichkeit, Berechnungen durchzuführen (Molekülgewicht usw.). Die aktuelle Entwicklerversion zeigt auch schon die Absorptions- und Emissionsspektren der Elemente an. Des Weiteren ist eine Tabellenansicht der Isotopen und ihrer Zerfallsarten in Arbeit. Die Homepage von Kalzium finden Sie unter *http://edu.kde.org/Kalzium*.



Abbildung 9.16: Kalzium äußert sich zu Eisen.

9.6.2 GENtle

GENtle ist eine Software für molekularbiologische Aufgaben wie das Bearbeiten und Verwalten von DNA- und Aminosäure-Sequenzen, zum Erstellen von Plasmidkarten, Protein-Analyse und verwandten Arbeiten. Es steht unter der GPL und besitzt eine deutsche Dokumentation.

Im DNS-Modul können DNS-Sequenzen angezeigt, annotiert und bearbeitet werden. Die Baumdarstellung zeigt Eigenschaften der DNS-Sequenz, der Features und der Restriktionsschnittstellen. Diese können per Doppelklick ein- bzw. ausgeblendet werden. Die Kartendarstellung zeigt eine Übersicht der gesamten DNS-Sequenz. Features und Restriktionsschnittstellen werden angezeigt. Zusätzlich zur DNS-Sequenz können komplementäre Sequenz, Features, Restriktionsschnittstellen, kodierte Aminosäuresequenzen und deren Protease-Schnittstellen dargestellt werden.

Das Peptid- oder Protein-Modul ermöglicht das Darstellen, Annotieren, Analysieren und Bearbeiten von Aminosäuresequenzen. Im PCR-Modul können Primer erstellt und „virtuelle PCRs“ durchgeführt werden.

Das PCR-Modul kann aus dem DNS-Modul gestartet werden. Wenn eine Sequenz markiert ist, können automatisch Primer am Anfang bzw. am Ende der markierten Sequenz erstellt werden. Ist die markierte Sequenz genau drei Nukleotide lang, lassen sich wahlweise auch Mutagenese-Primer erstellen.

Das Sequenzierungs-Modul kann Sequenzierungs-Dateien im ABI-Format lesen, darstellen und bietet die Möglichkeit, diese zu bearbeiten. Es werden Informationen über die Sequenzierung angezeigt. Im Editiermodus können Nukleotide ersetzt werden; ein Einfügen oder Entfernen von Nukleotiden ist nicht möglich.

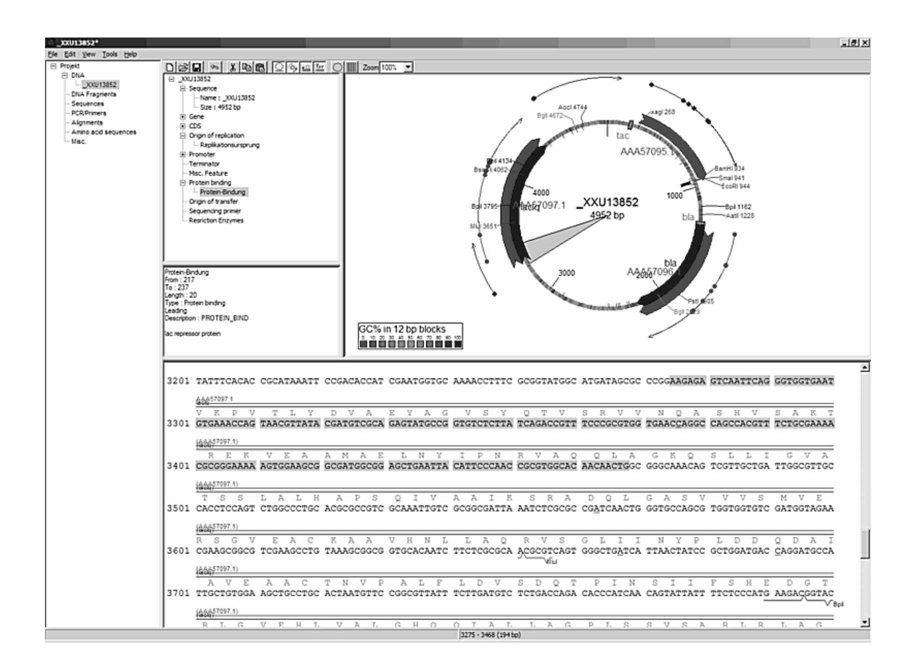


Abbildung 9.17: GENtle zeugt DNS-Sequenzen.

Das Alignment-Modul ermöglicht die Berechnung und Darstellung von DNS- und Aminosäure-Sequenz-Alignments. Als Algorithmen stehen zur Verfügung:

- Clustal-W: Dies ist der Default-Algorithmus. Er benutzt ein externes Programm zur Berechnung. Clustal-W erzeugt globale Alignments (Sequenzen, die etwa gleich lang sind) von hoher Qualität.
- Smith-Waterman: Ein interner Algorithmus für lokale Alignments (mehrere kurze Sequenzen werden gegen eine längere aligned). Dabei ist die erste Sequenz die lange Master-Sequenz, gegen die alle anderen Sequenzen aligned werden. Der Algorithmus ist schneller als Clustal-W, eignet sich aber nur für einfache Alignments, insbesondere zur Überprüfung von Sequenzierungen.
- Needlemann-Wunsch: Der Algorithmus ist schneller, aber qualitativ schlechter als Clustal-W. Wie bei Smith-Waterman werden alle Sequenzen gegen die erste Sequenz aligned, nicht untereinander. Der Algorithmus eignet sich zum schnellen Vergleich mehrerer, (sehr) ähnlicher Sequenzen.

Das Rechner-Modul umfasst mehrere spezialisierte Rechner für typische molekularbiologische Aufgaben. Der Rechner „Ligation“ ermittelt die zu pipettierenden Mengen von Insert und Vektor für eine Ligation. Der Rechner „DNS-Konzentration“ berechnet sus Photometerwerten der Absorption einer DNS-Lösung bei 260 bzw. 280 nm Reinheit und Menge der DNS. Der Rechner „Pro-

tein“ berechnet aus Photometerwerten der Absorption einer Protein-Lösung bei 250 bzw. 280 nm Protein-Reinheit und -Konzentration.

Das Bildbetrachter-Modul dient zur Darstellung von digitalen Bildern, vornehmlich Gel-Bildern. Unterstützt wird momentan das IMG-Format von BioRad. Über das Kontextmenü kann das Bild in die Zwischenablage kopiert, als BMP-Datei gespeichert oder ausgedruckt werden.

Des Weiteren verfügt das Programm noch über eine Datenbankverwaltung und diverse weitere Werkzeuge. Mehr zum Programm erfahren Sie unter *http://gentle.magnusmanske.de/*.

RasMol

RasMol ist ein Werkzeug zur grafischen Visualisierung von Proteinen, Nukleinsäuren und kleineren Molekülen. Die Moleküle können auf verschiedenste Art dreidimensional dargestellt und in Echtzeit gedreht werden. Zu den Darstellungsmöglichkeiten gehören Kalottenmodell, Stabmodell und andere. Die Bedienung des Programmes ist etwas ungewöhnlich. Man benötigt nämlich eine Textkonsole, von der aus man auch das Programm gestartet hat. Hier erfolgen die interaktiven Eingaben. Das Programm ist sehr ausführlich dokumentiert. Die Moleküldarstellungen können in die verschiedensten Grafikformate exportiert werden.

RasMol wird von der Pharmafirma Glaxo in Zusammenarbeit mit der Universität von Edinburgh entwickelt. Maintainer ist Roger Sayle von Glaxo Research and Development. RasMol kann auch dreidimensionale Protein-Strukturen abbilden, die im Format der „Brookhaven Protein Database“ (*.pdb, *.ent) vorliegen. Verschiedene andere Formate wie *.mdl, *.mol2 und *.xyz sind ebenfalls ladbar.

Angefangen hat RasMol vor etwa zehn Jahren mit einem Grafikfenster, das aber interaktiv per Kommando-Eingabe gesteuert wurde. So verfügt RasMol seither über eine recht mächtige Script-Sprache, die jedoch eher selten genutzt wird. Heutzutage werden von den Benutzern meist nur die im grafischen Menü zur Verfügung gestellten Optionen benutzt, weshalb sich die Menüstruktur von Version zu Version erweitert. Außer den Hauptmenüpunkten gibt es die Menüs *Farbliche Gestaltung*, *Auswahl 1*, *Auswahl 2*, *Aminosäuren* und *Makros*. Sie alle nutzen das Konzept von RasMol: Die Kommandos beziehen sich immer auf eine ausgewählte Anzahl von Atomen. Also ist immer der erste Schritt, per Befehl Atome anhand eines Kriteriums auszuwählen und dann in den folgenden Schritten zu bearbeiten. In der Script-Sprache wird dieses durch den Befehl *select* (bzw. *restrict*) erreicht. Danach beziehen sich alle weiteren Kommandos auf diese ausgewählten Atome. Ein solcher Befehl könnte beispielsweise *spacefill on* oder *color yellow* sein. RasMol hält auch eine Menge vordefinierter Selektions-Befehle parat. Viele dieser Befehle beziehen sich auf die Eigenschaften von Aminosäuren wie zum Beispiel aliphatisch oder polar. Andere beziehen sich auf abgeschlossene Substrukturen wie Wasser, Ionen oder einen Liganden. Mit dem

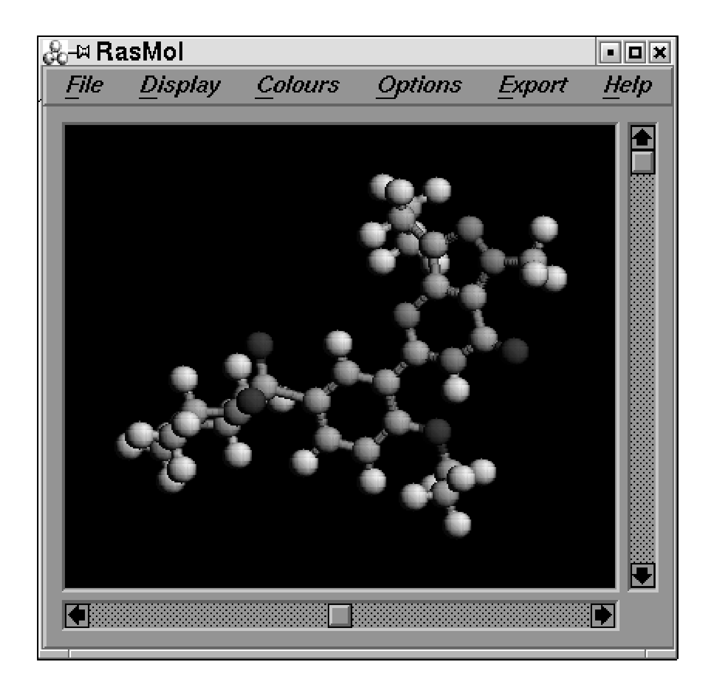


Abbildung 9.18: Molekül, mit Rasmol erzeugt

Menü *Aminosäuren* ist es möglich, sich bestimmte Aminosäuren in dem Protein anzeigen zu lassen. Die Abfolge der Aminosäuren in diesem Menü ist durch den Ein-Buchstaben-Code gekennzeichnet.

Sie haben also die Möglichkeit, auch mit Untermengen eines Moleküls zu arbeiten und dies mit der gesamten Fülle der Aktionen, die RasMol dafür zur Verfügung stellt. Sie können so nacheinander verschiedene Untermengen auswählen und bearbeiten. Durch diese Möglichkeit ist es auch Nicht-Fachleuten möglich, recht aussagekräftige Bilder zu erzeugen. Diese können von RasMol in etlichen Formaten abgespeichert werden (gif, bmp, ps etc.).

Mit dem Makro-Menü lassen sich Dateien einbinden, die RasMol-Befehle enthalten. Hat man zum Beispiel immer wiederkehrende Befehlsabfolgen, so erzeugt man eine Datei mit der Endung „.scr“ und legt diese in einem spezielles Verzeichnis ab. Beim Start durchsucht RasMol dieses Verzeichnis nach vorhandenen Makros und lädt diese.

Viele weitere Informationen, ein Handbuch und natürlich das Programm finden Sie unter *http://www.openRasMol.org*.

9.7 Elektrotechnik

9.7.1 KRelais

Wir wollen zur Einstimmung mit einem netten kleinen Tool zur Simulation von Relais-Schaltungen beginnen. (Die getestete Version vom März 2006 trägt die Nummer 0.2.) Auf einer Zeichenfläche sind sieben Zeilen mit jeweils zehn Plätzen (7 x 10 Knoten einschließlich Stromversorgung und Masse) vorhanden. Die passiven Bauelemente können in den ersten fünf Zeilen platziert werden. In der sechsten Zeile dürfen nur aktive Bauteile wie Relais und Lampen untergebracht werden. An Bauteilen gibt es: Taster (Schließer und Öffner), Schalter, Relaiskontakt (Schließer und Öffner), Relais, SE-Relais (t*100 ms), SI-Relais (t*100 ms) und eine Lampe.

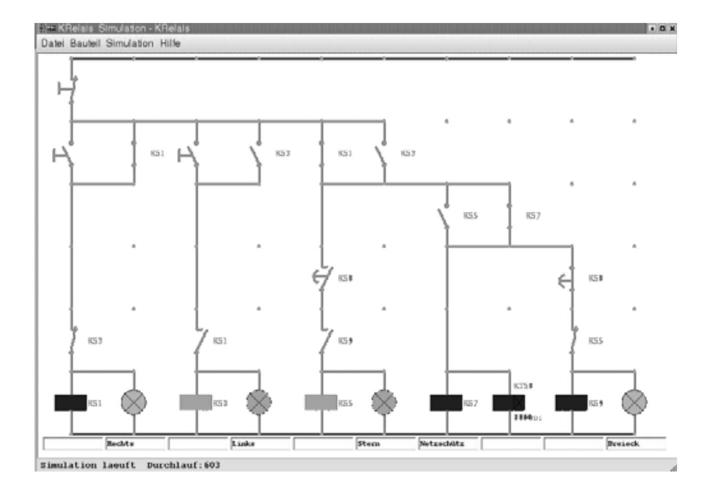


Abbildung 9.19: KRelais ist echt einfach aufgebaut.

Die Bauteile werden per Menü ausgewählt und durch Klicken mit der linken Maustaste zwischen zwei Knoten installiert und durch Anklicken mit der rechten Maustaste wieder gelöscht. Sie lassen sich horizontal und vertikal verbinden. Relais werden nach ihrer Lage automatisch nach Zeile und Spalte nummeriert (z. B. K43). Den SE- und SI-Relais wird eine Verzögerungszeit zugeordnet, die in Einheiten von 100 ms gezählt wird. Taster und Schalter sind passive Bauelemente, die Relais und Lampen schalten können. Sie werden mit einem Relais verbunden, indem man die Nummer des Relais eingibt.

Nach dem Platzieren und Verbinden von Kontakten, Schaltern und Relais kann die Simulation gestartet werden. Schalter und Taster lassen sich während der Simulation mit der linken Maustaste schalten. Die Kontakte eines Relais, SE-Relais oder SI-Relais werden über die Nummer dem jeweiligen Relais zugeordnet (zum Relais K45 gehören also alle Kontakte mit der Nummer K45).

Weitere Informationen und natürlich die aktuelle Programmversion finden Sie unter *http://www.technische-simulation.de/*.

9.7.2 KLogic

KLogic ist eine Anwendung zum einfachen Entwurf und zur Simulation von Digitalschaltungen. Das Programm ermöglicht den einfachen Aufbau von Schaltungen aus Standardkomponenten wie UND, ODER oder XOR. Als Speicherelemente stehen RS-Flipflops und JK-Flipflops zur Verfügung. Um komplexere und wiederverwendbare Schaltungen zu entwerfen, können Teilschaltungen gespeichert und wieder geladen werden. DIe Software unterliegt der GPL.

Die Simulation erfolgt standardmäßig bereits während des Entwurfs einer Schaltung. Zur erweiterten Schaltungsanalyse können einzelne Simulationsschritte ausgeführt werden. Das bietet die Möglichkeit, kurzzeitige undefinierte Schaltungszustände zu untersuchen. Der Signalverlauf der Schaltung lässt sich als Graph darstellen. Jedes Bauteil einer Schaltung hat eine einstellbare (Durgangs-)Verzögerung. Mit der Burst-Option schaltet man die Verzögerungen aller Bauteile aus.

Das Programm besitzt eine CAD-Oberfläche, in der man die einzelnen Bauteile platziert und miteinander verbindet. Das läuft ab, wie in fast jedem Mal- oder Konstruktionsprogramm. Neben den Logikschaltkreisen gibt es noch Eingänge, die jeweils mit logisch 1 oder 0 belegt werden können. Als Ausgänge dienen LEDs (für einfache Anwendungen) oder man kann sich ein komplettes Impulsdiagramm ausgeben lassen. Erzeugt werden die Ausgabedaten von der Simulation der Schaltung. Zudem lässt sich die Logikgleichungen zur gezeichneten Schaltung ausgeben.

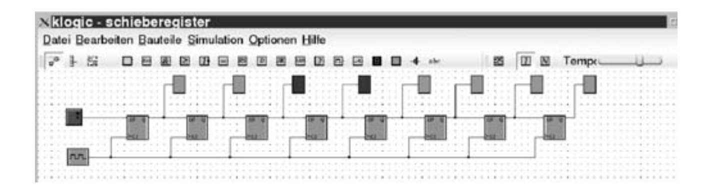


Abbildung 9.20: Ein Schieberegister, simuliert mit Klogic

Die Simulation läuft getaktet ab. Nach jedem Simulationsschritt werden alle Ausgangszustände ermittelt und ausgegeben. Die Taktrate kann per Schieberegler verändert werden. Wie in Wirklichkeit haben alle Schaltglieder eine Durchlaufzeit, so dass – in begremztem Umfang – auch Anomalien (Hazards) untersucht werden können. Der Einzelschrittbetrieb erlaubt Ihnen, das Verhalten einer Schaltung, beispielsweise eines Zählers, Schritt für Schritt zu simulieren. Eine bei der Fehlersuche hilfreiche Funktion! Zusätzlich kann man auch

das zugehörige KV-Diagramm ausgeben (aber bitte nur bei wenigen Variablen, denn in sehr komplexen Fällen ist das Programm nicht immer stabil.

Über die Basisfunktionen hinausgehende Features sind neben der schon erwähnten Möglichkeit, Schaltungsblöcke zusammenzufassen und für spätere Verwendung abzuspeichern, die so genannten Equatation Devices. Für malfaule Benutzer bieten sie die Möglichkeit, Logikgleichungen direkt einzugeben und Komponenten zu definieren, die ein komplettes Schaltnetz enthalten. Die Gleichungen dürfen auch geklammerte logische Ausdrücke enthalten. Auf diese Weise kann man komplexere Schaltungen übersichtlich halten. Als Zuckerstückchen bietet KLogic auch noch ein kleines 32 x 8 RAM.

Download, Screenshots und Tutorial finden Sie unter *http://www.a-rostin.de/klogic/indexd.html*.

9.7.3 KTechlab

Obwohl es sich bei KTechlab noch um eine recht junge Anwendung handelt (Versionsnummer 0.3), bietet das Tool Elektro-Ingenieuren bereits eine Menge von Funktionen zur Entwicklung und Simulation von elektrischen Schaltkreisen. Es lassen sich mit KTechlab komplette digitale Schaltungen entwerfen und simulieren. Es werden auf der digitalen Ebene Gatter, Inverter, Buffer, Flipflops, Zähler, Addierer, Multiplexer und Anzeigen unterstützt. Weiterhin gibt es frei konfigurierbare RAMs und Komparatoren. Die Zahl der Gattereingänge ist beliebig justierbar. Auf der analogen Seite können Sie Analog-Digital- und Digital-Analog-Wandler anschließen. Die Logiksimulation erlaubt bis zu 1 Mio. Schritte pro Sekunde (wenn die Rechner-Hardware mitspielt). Es lassen sich sogar die parallele und die serielle Schnittstelle des PC einbinden.

Neben den Digitalschaltungen erstellt das Programm auch Flussdiagrame für PIC-Chips (eine Familie von RISC-Prozessoren der Firma Microchip). Die SImulation erfolgt hier mithilfe des externen Programms `gpsim`. DIe Programmierung der PICs kann auf verschiedene Arten erfolgen:

- Die Programmiersprache „Microbe“ ist eine Sprache mit Elementen aus C und Basic.
- FlowCode erlaubt die Programmierung mit Flussdiagrammen. Die Diagramme werden dann in Microbe, Assembler oder Binärcode umgesetzt.
- KTechlab erlaubt mittels der `gputils` auch Assembler-Programmierung der PIC-Prozessoren.
- Durch die Integration des SDCC lassen sich C-Programme für die PICs erstellen.

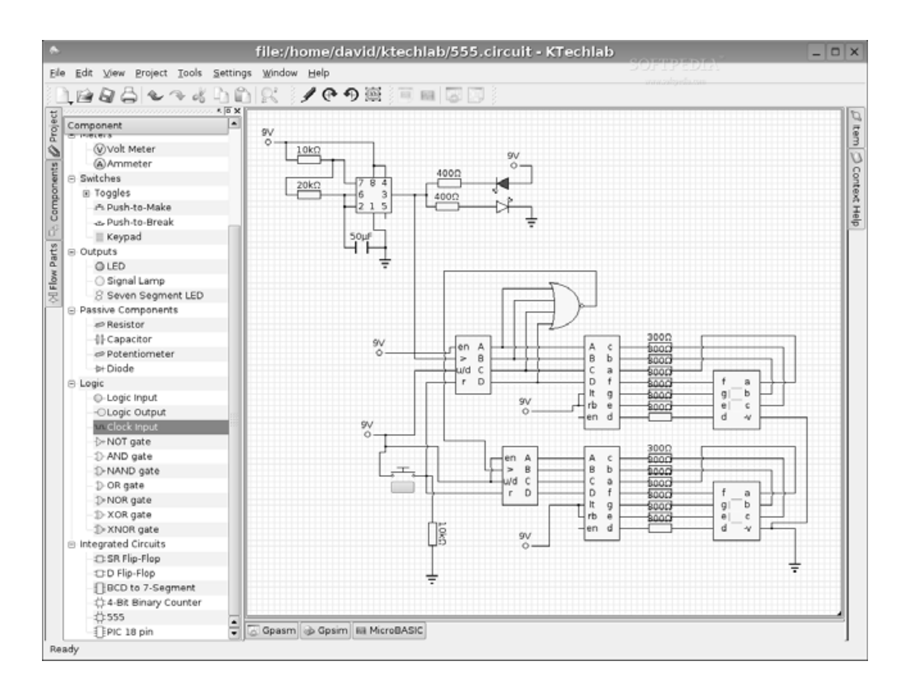


Abbildung 9.21: Schaltung simuliert mit KTechlab

Auf der analogen Seite erlaubt das Programm die Simulation von Spannungsquellen, verschiedenen Signalquellen sowie Transistoren (NPN and PNP), Dioden, passive Bauelementen (R, C, L), Operationsverstärkern (ideal), Schaltern und Anzeige-Elementen (Glühlampe, LED, 7-Segment-Display, Punktmatrix-Display). Schließlich gibt es noch den Timerbaustein NE555. Als universelles Messinstrument ist ein Oszilloskop in die Simulation integriert.

Eine ausführliche Installationsanleitung sowie eine kurze Einführung in die bisher verfügbaren Funkionen und die geplanten Features gibt es auf der KTechlab-Webseite. Hier finden Sie auch Links zu den Sourcen und den ebenfalls benötigten Bibliotheken `gpsim` und `gputils`: *http://ktechlab.org/*.

9.7.4 Chipmunk

Das Chipmunk-Paket ist eine Entwicklung des California Institute of Technology. Eigentliche Aufgabe des Paketes ist der Entwurf von ASICs. Für die meisten Anwender dürfte hauptsächlich der Simulator „log“ von Interesse sein. Er erlaubt die grafische Eingabe der Stromlaufpläne und die Simulation von Digital- und Analogschaltungen. Für genauere Simulationen können Ausgaben für „Spice“ generiert werden.

Das Chipmunk-Paket enthält mehrere Werkzeuge zur Entwicklung und Simulation elektronischer Schaltungen. „Log“ unterstützt die Erstellung von Schaltplänen für Dokumentationszwecke und erzeugt Netzlisten im SPICE- und NTK-Format. Schaltpläne landen auf Wunsch nicht nur auf dem Bildschirm, sondern auch als PostScript- oder HP-GL-Datei auf der Festplatte.

Die Schaltungssimulatoren „Analog“ von John Lazzaro und „Diglog“ von Dave Gillespie erlauben die Simulation und Erstellung von Schaltungen. Die Bauteile-Bibliothek reicht von Schaltern über Leuchtdioden bis hin zu einer ganzen Reihe von TTL-Bausteinen der 7400-er Serie.

Chipmunk unterliegt der Gnu Public Licence (GPL) und wird nur im Quellcode zum Download angeboten. Die C-Compiler verschiedener UNIX-Derivate kompilieren das Programmpaket problemlos. Zum kompletten System gehören neben „Log“ die Pakete „View“, „Until“ und „Wol“. Das Webdoc-Paket beschreibt alle Programme ausführlich. Zu finden ist alles unter *http://www.pcmp.caltech.edu/chipmunk/*, wobei die Site manchmal schwer erreichbar ist.

9.7.5 FreeHDL

FreeHDL ist ein Simulator für VHDL, welcher zum VHDL-93-Standard kompatibel ist. VHDL ist eine Hardware-Beschreibungssprache, mit der man Schaltkreise entwerfen kann. Momentan gibt es keine freie vollständige VHDL-Implementierung. Alle freien Tools unterstützen nur strukturelles VHDL. Ziel des FreeHDL-Projektes ist die Erstellung eines freien VHDL-Compiler- und -Simulatorsystems, das den kompletten VHDL-93-Standard unterstützt. Ergänzt wird das Sytem durch einen Source Level Debugger und einen Waveform-Viewer.

An dieser Stelle soll ein Beispiel verdeutlichen, was man mit VHDL anfangen kann. Es soll ein Vorwärts-Rückwärts-Zähler mit 4 Bit Wortbreite entwickelt werden. Der Zähler hat somit vier Datenausgänge (C0, C1, C2, C3), einen Reset-Eingang (Reset = 0 setzt den Zähler auf 0), einen Takteingang (Clock) und einen Eingang zur Festlegung der Zählrichtung (1 = Up, 0 = Down).

Der vollständige VHDL-Code des Zählers sieht einem C-Programm recht ähnlich. Um diesen dann auf seine vollständige und korrekte Implementierung zu prüfen, ist eine Simulation notwendig, für die man wieder ein VHDL-Programm schreiben muß. Das Listing des Zählers soll hier aber ausreichen:

```
library IEEE;
use IEEE.std_logic_1164.all; -- import std_logic types
 use IEEE.std_logic_arith.all; -- import add/sub of std_logic_vector

ENTITY counter_updown is
PORT ( -- Counter Control Signals
        clock          : IN  std_logic;  -- counter clock
        reset          : IN  std_logic;  -- counter reset
        updown         : IN  std_logic;  -- count direction up/down
```

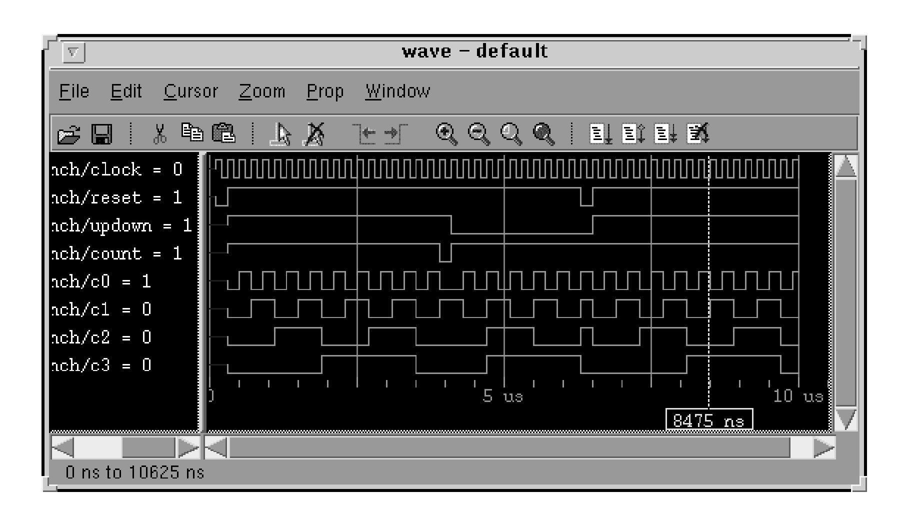


Abbildung 9.22: 4-Bit-Zähler im Waveform-Viewer von FreeHDL

```
        count           : IN  std_logic;  -- count enable
       -- Counter Output Signals
        c0, c1, c2, c3  : OUT std_logic); -- counter bit 0..3
END counter_updown;

ARCHITECTURE rtl OF counter_updown IS
  -- internal signal declaration
  SIGNAL counter : std_logic_vector(3 DOWNTO 0);
   BEGIN

   count_process : PROCESS
   -- This process builds the counter
   BEGIN
    -- wait for rising edge of clock (0->1)
   WAIT UNTIL clock'EVENT AND clock = '1';
   IF reset = '0' THEN
     -- set all counter bits to zero
      counter <= (others => '0');
   ELSE
     IF (count = '1') AND (updown = '1') THEN
       -- count up: n, n+1, n+2, ..., 15, 0, 1, ...
        counter <= CONV_STD_LOGIC_VECTOR(UNSIGNED(counter) + 1, 4);
     ELSIF (count = '1') AND (updown = '0') THEN
        -- count down: n, n-1, n-2, ..., 0, 15, 14, ...
       counter <= CONV_STD_LOGIC_VECTOR(UNSIGNED(counter) - 1, 4);
     ELSE
        -- hold counter in the same value as before
       counter <= counter;
     END IF;
```

```
  END IF;
  END PROCESS;
  -- Connect internal counter to outputs
  c0 <= counter(0);
  c1 <= counter(1);
  c2 <= counter(2);
  c3 <= counter(3);
END rtl;
```

Das Bild zeigt das Ergebnis der Simulation. Aus dem Listing wird dann eine Schaltung generiert, die man real in eine ASIC umsetzen könnte. Dazu ist dann noch ein Layoutprozess nötig.

FreeHDL kann von *http://www.freehdl.seul.org/* heruntergeladen werden.

9.7.6 Electric

Electric ist ein EDA-Tool, das bereits seit 1988 existiert. Bisher galt, dass Programme zum Design integrierter Schaltungen zumeist ein Problem hatten – sie waren entweder sehr teuer oder besaßen nicht die benötigten Fähigkeiten. Electric wird unter der GPL vertrieben und ist daher kostenlos. Es unterstützt sowohl IC-Layout als auch schematisches Design. Dabei können textbasierte Sprachen zur Anwendung kommen (VHDL) und der Nutzer kann seine Chips „programmieren“. Neben der Fähigkeit, komplexe geometrische Rahmenbedingungen festzulegen, macht es den User außerdem größtenteils unabhängig vom verwendeten System, da es die gängigen Dateiformate versteht und auch unter Windows läuft.

Als Electric zum GNU-Projekt wurde, hatte es bereits eine über 15-jährige Entwicklung hinter sich. Begonnen hat es in Schlumbergers PaloAlto-Forschungslabor, wurde zwischenzeitlich kommerziell vertrieben und ist nun freie Software. Electric unterstützt neben bipolaren die NMOS- und CMOS-Technologien und digitale Filter. Es kann unter anderem folgende Dateiformate einlesen: CIF, DXF, VHDL, HPGL, QuickDraw, PostScript und bedingt EDIF, SDF sowie Verilog. Einzigartig unter den freien Tools ist der Silicon Compiler, der nahezu vollautomatisch aus einem Schematic VHDL generiert und dieses vollautomatisch routet.

Herunterladen kann man das Programm unter *http://www.electriceditor.com.* Dort findet man auch Anleitungen und alle nötigen Informationen.

9.7.7 Eagle

Bisher war nur von Simulation und Schaltplänen die Rede. Will man aber eine reale Schaltung aufbauen, braucht man neben dem Schaltplan auch ein Layout für eine gedruckte Schaltung („Platine“ genannt) – zumindest dann, wenn es

nicht nur um einen Prototypen auf Lochrasterplatte geht. Eagle gibt es für diverse Betriebssysteme und es hat inzwischen einen hohen Marktanteil. Es ist auch das einzige kommerzielle Produkt unter den vorgestellten Programmen. Es gibt aber auch eine Freeware-Version mit Limitationen, aber diese sind für den Einsatz im Studium nicht weiter störend: Platinen dürfen nicht größer als 100 mal 80 mm sein, es gibt nur eine Seite für den Schaltplan und nur zwei Layer. Aber als Normalsterblicher kann man sowieso nicht mehr als zwei Layer belichten, geschweige denn ätzen. Meist entwickelt man privat oder als Student nur einseitige Platinen und so ist dieses Handikap eigentlich keines mehr. Betonen sollte man, dass auch in der Light-Version der Autorouter funktioniert. Neben dem Programm kann man sich ebenfalls kostenlos die Eagle zugehörigen Bauteile-Bibliotheken herunterladen. Wem diese nicht reichen, der wird auf dem Webserver von CadSoft fündig. Des Weiteren wird ein HTML-basiertes Hilfesystem, das jeden Befehl von Eagle beschreibt, in Englisch und Deutsch mitgeliefert.

Die große Zahl von Bauteile-Bibliotheken und der exzellente Autorouter sind die beiden größten Stärken von Eagle. Wie bei jedem Programm sind die Electrical Rule Check und Design Rule Check eingebaut. Diese beiden Module prüfen den Schaltplan auf Einhaltung der Regeln für die Verschaltung der Bauteile und die Platine auf die Verlegung der Leiterbahnen. Für Neulinge ist die Einführung auf der Homepage des Herstellers sehr hilfreich, obwohl die Bedienung des Programms sehr intuitiv gehalten ist. Angenehm fällt das komfortable Controlcenter auf. Hier können alle Projekte übersichtlich verwaltet und eingesehen werden. Neben Schaltplan und Platinenlayout (auch getrennt für jedes Layer) lassen sich auch noch Bohrschabone und Bestückungsplan ausgeben. Letzterer lässt sich fast so behandeln wie ein eigenes Layer und durch Zeichnungen bzw Beschriftungen ergänzen. Ein Export der Eagle-Daten in dieverse andere Formate ist möglich. Viele Platinenhersteller können die Eagle-Daten aber auch direkt verarbeiten.

Ein interessantes Feature für Profis ist die Möglichkeit, eigene Bauteile-Bibliotheken zu erstellen. Dabei wird nicht nur das Schaltzeichen festgelegt, sondern auch das Aussehen des realen Bauteils für den Bestückungsplan der Platine. Das „Malen“ eines neuen Bauteils geht recht fix, so dass man schnell seine eigenen Spezialbauteile ins Programm integriert hat.

Ein Export der Eagle-Daten in diverse andere Formate ist möglich. Viele Platinenhersteller sind auch in der Lage, die Eagle-Daten direkt zu verarbeiten. Natürlich können Sie die entwickelte Schaltung und das Platinenlayout auf dem Drucker oder einem anderen Gerät ausgeben. Gerade hier ist Maßarbeit gefordert, denn was nützt einem das schönste Platinenlayout, wenn man es nicht auf Papier oder Folie (zum Belichten) bekommt. CadSoft klassifiziert die Ausgabemedien wie folgt: Matrixdrucker, Laserdrucker, PostScript, Stift-Plotter, Gerber Plotter, Excellon und Sieb&Meyer-Bohrdateien sowie Bitmap-Dateien.

Die grafische Oberfläche ist unter Zuhilfenahme des GUI-Interfaces Zinc realisiert und schon auf den ersten Blick ansprechend. Zum Zeichnen der Schaltung

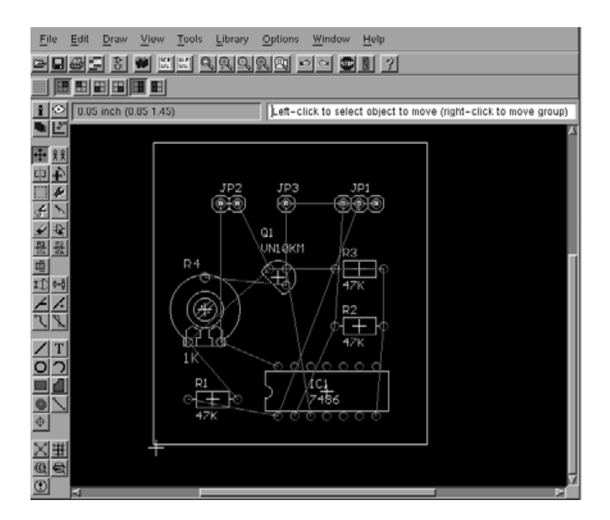


Abbildung 9.23: Bauteile in Eagle platziert

nimmt man sich die entsprechenden Bauteile aus der Bibliothek und platziert das Bauteil-Symbol auf dem Schaltplan. Mit der rechten Maustaste kann man die Schalt-Symbole drehen. Anschließend verbindet man die Bauteile miteinander. Zu jeder Zeit lassen sich Elemente aus der Bibliothek nachladen oder löschen. Bei den Leitungen ist zu beachten, dass eine Linienkreuzung keine Verbindung der Leitungen darstellt. Zu diesem Zweck muss man noch Verbindungen (Junctions) setzen. Man sollte für den Anschluss an die Stomversorgung umbedingt ein Bauteil aus der Supply-Bibliothek verwenden. Wenn Bauteile verwendet werden, die implizit angeschlossene Versorgungspins besitzen, muss auch ein passendes Supply-Symbol vorhanden sein.

Von dem Schaltplan-Editor aus wechselt man durch einen Klick auf das Board-Symbol in die Layoutfunktion. Der Board-Editor erzeugt die Bauteile und Leitungen als Gummifäden und legt sie neben der Platine ab. Existiert schon ein Board zu einem Schaltschema, so wird dieses geladen, anstatt eines neu aufzubauen. Danach beginnt das Platinendesign, indem Sie durch Umordnen oder Drehen von Bauteilen eine Vorgabe für den Autorouter erzeugen. Sie werden natürlich versuchen, beim Platzieren einen günstigen Kompromiss zwischen mechanischen, elektrischen, thermischen und elektromagnetischen Anforderungen zu finden. Günstige Voraussetzungen für den späteren Route-Vorgang sind dann geschaffen, wenn die Gummibänder möglichst kurz sind und sich nur wenige von ihnen kreuzen. Bei mehrfach vorhandenen Funktionsgruppen innerhalb eines Bauteils (zum Beispiel bei Treiber-ICs) ist es möglich, diese gegeneinander zu tauschen um Vorteile bei der Platzierung zu erzielen.

Von diesen Änderungen muss das Schaltplan-Modul natürlich informiert werden. Daher hat Eagle als wichtiges Ausstattungsmerkmal die Back- und Forward-

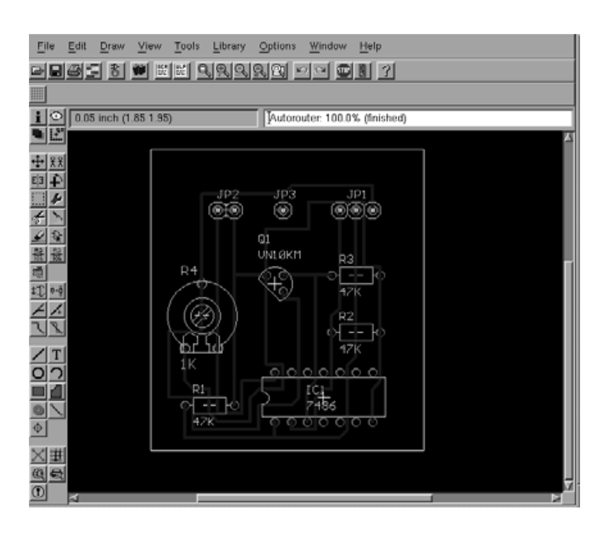


Abbildung 9.24: Das fertige Eagle-Layout

Annotation. Diese Automatik hält Schaltplan- und Layout-Files auf einem Stand. Sind schließlich alle Bauteile gut platziert, kann man dem Autorouter das Verlegen der Leitungsbahnen anvertrauen. Eine manuelle Nacharbeit wird jedoch fast immer nötig sein.

Eagle für alle Plattformen, Bibliotheken und weitere Infos finden Sie unter *http://www.cadsoft.de*.

9.8 Datenbanken

9.8.1 Desktop-Wiki mit Zim

Wikis werden nicht nur im Web immer beliebter. Der WYSIWYG-Editor Zim transportiert das Web-Konzept auf den Desktop. Das Programm ist in Perl geschrieben und bietet alle Grundfunktionen der Textformatierung. Bemerkenswert ist die Möglichkeit, Begriffe mit anderen Einträgen zu verlinken. Existiert der zugehörige Beitrag noch nicht, legt Zim ihn automatisch an. Eintragsseiten lassen sich natürlich auch wieder entfernen oder an eine andere Stelle verschieben.

Das Programm eignet sich hervorragend, um eigene Notizen oder To-Do-Listen zu verwalten. Aber man kann auch ein privates System von Manual-Seiten oder Ähnliches aufbauen. Zim ist ein relativ junges Projekt (es wurde im September 2005 begonnen und die Versionsnummer stand Anfang 2006 gerade mal bei 0.12), wirkt jedoch schon recht ansprechend. Man kann gespannt sein, was sich daraus

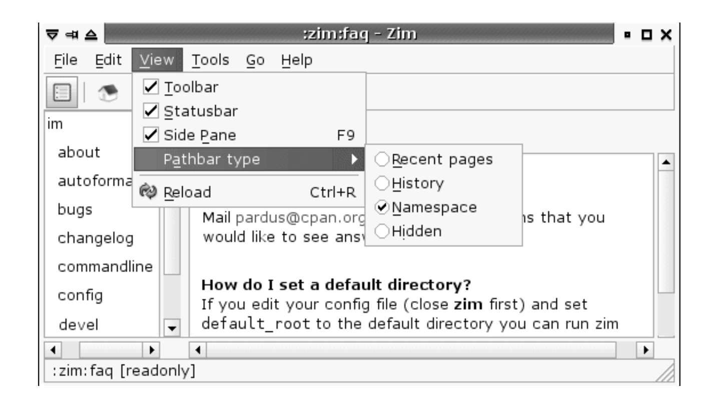


Abbildung 9.25: Zim, das Desktop-Wicki

entwickelt. Da es in Perl geschrieben wurde, sind natürlich Perl und die entsprechenden Module (hauptsächlich GTK) notwendig. Weitere Informationen und die neueste Version finden Sie unter *http://zoidberg.student.utwente.nl/zim/*.

9.8.2 MySQL

MySQL (sprich „Mai Es Ku El") ist ein relationales Datenbanksystem (RDBMS = Relational Data Base Management System) der Firma MySQL AB. MySQL ist gleichzeitig die populärste Open-Source-Datenbank. MySQL AB ist ein kommerzielles Unternehmen, dessen Geschäft darin besteht, Serviceleistungen rund um die MySQL-Datenbank zur Verfügung zu stellen. Mit MySQL lässt sich auf einer Vielzahl von Plattformen ein Datenbank-Server einrichten, der multitasking- und multiuser-fähig ist. Da die Welt der Datenbanken nicht jedem geläufig ist, wollen wir zunächst ein paar Begriffe klären:

Eine **Datenbank** ist eine strukturierte Sammlung von Daten. Das kann alles mögliche sein – von einer einfachen Einkaufsliste über eine Bildergalerie bis zu riesigen Informationsmengen in einem Unternehmensnetzwerk. Um Daten zu einer Computer-Datenbank hinzuzufügen, auf sie zuzugreifen und sie zu verarbeiten, benötigen Sie ein Datenbank-Managementsystem wie MySQL.

SQL ist eine von IBM entwickelte standardisierte Datenbanksprache, die das Speichern, Aktualisieren und den Zugriff auf Informationen erleichtert (SQL = Structured Query Language).

Eine **relationale Datenbank** speichert Daten in separaten Tabellen, anstatt sie alle in einem einzigen großen Speicherraum unterzubringen. Hierdurch werden hohe Geschwindigkeit und Flexibilität erreicht. Die Tabellen werden durch definierte Beziehungen verbunden (Relationen), was es möglich macht, Daten aus verschiedenen Tabellen auf Nachfrage zu kombinieren.

Zur Erfassung von Daten dienen oft Formulare. Sie bestehen aus einzelnen Datenfeldern mit einem Feldnamen, der das einzutragende Datum beschreibt. Für jedes Datenfeld steht ein bestimmter Platz zur Verfügung. Der Datentyp gibt an, ob numerische Daten, Texte oder andere Dinge eingegeben werden. Zusammengehörende Daten fasst man in einem Datensatz zusammen. Bei einer Adress-Datenbank gibt es beispielsweise Felder wie Vorname, Name, Straße, Postleitzahl, Wohnort usw. Eine Adresse bildet somit einen Datensatz. Alle Datensätze zusammen bilden eine Tabelle (table). Zwischen mehreren Tabellen kann es Beziehungen geben, so dass man alle zusammengehörenden Tabellen in einer Datenbank zusammenfasst. Eine Tabelle lässt sich nach einem Schlüssel oder mehreren Schlüsseln sortieren. Die Ausgabe und die Suche von Daten kann nicht nur auf bestimmte Datenfelder eingeschränkt werden, sondern auch mit Bedingungen verknüpft. Dann werden nur solche Datenfelder ausgegeben, die der Bedingung genügen, z. B. alle Adressen aus Bielefeld.

MySQL wird manchmal mit SQL verwechselt. MySQL ist nicht eine Art SQL, sondern ein Datenbanksystem, das SQL benutzt, um Daten zu bearbeiten. SQL ist eine Programmiersprache, die von MySQL genutzt wird, um Arbeiten innerhalb einer Datenbank zu verrichten. MySQL ist sehr schnell, zuverlässig und leicht zu benutzen. MySQL besitzt eine ganze Reihe praktischer Features, die in enger Kooperation mit seinen Benutzern entwickelt wurden. Es wurde ursprünglich entwickelt, um sehr große Datenbanken handhaben zu können, und zwar sehr viel schneller als damals existierende Lösungen. Heutzutage bietet MySQL eine umfangreiche Reihe sehr nützlicher Funktionen. Vielfach ist die zu speichernde Datenmenge aber gar nicht so groß (von der Geschwindigkeit würde eine Textdatei reichen, die ggf. mittels `grep` durchsucht wird) – wozu dann eine Datenbank? Der Nutzen liegt in diesem Fall in der Strukturierungsmöglichkeit der Daten in verschiedene Tabellen und in der Möglichkeit, bei einer Abfrage mehrere Tabellen zu kombinieren oder Zeilen und Spalten herauszuschneiden. Ein weiterer Vorteil liegt darin, dass Sie jederzeit neue Felder hinzufügen oder vorhandene Felder löschen können.

Die Felder selbst können die unterschiedlichsten Datentypen haben. Neben anderen kennt MySQL Integer (ganze Zahlen), Float (Fließkommazahlen), Decimal (Dezimalzahlen mit fester Anzahl von Nachkommastellen), Text, Varchar (Zeichenketten), Date (Datumsangaben) oder Blob (beliebige Binärobjekte).

Um Ihre Datenbank benutzen zu können, muss der MySQL-Server laufen, denn die Datenbank basiert auf einem Client-Server-Konzept (auch wenn alles auf einem Rechner läuft). Der Vorteil daran liegt in der universellen Ansprechbarkeit des Datenbanksystems. Das kann entweder über den Kommandozeilen-Client `mysql` erfolgen, aber auch über einen grafischen CLient oder aus Programmen heraus (C, Perl, PHP etc.). Damit ist die Datenbank auch für Webanwendungen nutzbar. Zum Starten des Servers unter Linux dient das Programm `mysqld_safe`, das aber normalerweise nicht von Hand aufgerufen wird (`mysqld_safe &`), sondern in einem der Start-Scripten in `/etc/init.d` (früher hieß es übrigens safe_mysqld). Es gibt mehrere Möglichkeiten, nachzuprüfen, ob der MySQL-

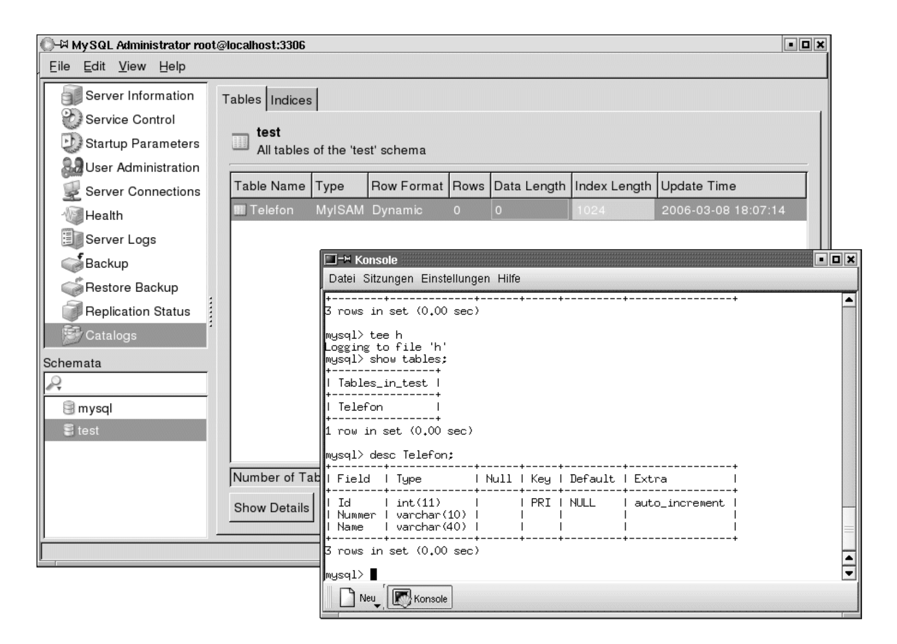


Abbildung 9.26: MySQL-Admin, das grafisch orientierte Tool und das mysql-Kommando im Shell-Fenster ergänzen sich prächtig.

Server läuft. Versuchen Sie es auf der Kommandozeilenebene mit `mysqladmin ping`. Normalerweise erhalten Sie die Antwort „mysqld is alive“.

Der `mysqld_safe`-Befehl ist das beste Mittel, um MySQL hochzufahren. Stürzt der Datenbankserver ab, so wird er automatisch neu gestartet. Zum Herunterfahren benutzen Sie den Befehl `mysqladmin shutdown` (wenn das nicht auch per Automatismus erledigt wird).

MySQL hat übrigens seine eigene Benutzerverwaltung, in der die Zugriffsrechte der Datenbankuser sehr subtil festgelegt werden können. Der User, der alles darf, heißt auch hier wieder „root“. Nach der Installation hat root noch kein Passwort, was bedeutet, dass momentan Jeder uneingeschränkten Zugriff auf die Datenbanken hat. Setzen Sie also ein root-Passwort:

mysqladmin -u root password top.secret

Danach müssen alle `mysql-admin`-Kommandos durch den Parameter `-p` ergänzt werden (also z. B. `mysqladmin -p ping`) und es wird nach dem Passwort von root gefragt.

Nach der Installation gibt es auch schon zwei Datenbanken, *mysql* für die Verwaltung der internen Daten und Benutzer und *test* zum Rumspielen. Wenn Sie sich das Verzeichnis der MySQL-Programme ansehen (bei Debian `/usr/bin/`),

finden Sie dort ca. drei Dutzend Programme, deren Namen mit „mysql" beginnen. Nachdem hier der Platz für eine Kompletteinführung in MySQL nicht ausreicht, bescheiden wir uns auf ein einziges Kommando für die Einführung in den Gebrauch der Datenbank, `mysql`. Das ist der Client auf der Kommandozeile. Natürlich gibt es auch grafisch orientierte Tools, die alles leisten, was Ihr Herz begehrt. Es sind dies:

- Mit dem MySQL-Administrator (`mysql-admin` können Sie alle Verwaltungsarbeit erledigen und auch neue Datenbanken anlegen.
- Mit dem Query-Browser (`mysql-query-browser`) lassen sich Datenbankabfragen aller Art komfortabel erledigen.

Jedes Mal wenn man `mysql` aufruft, muss man sich erst anmelden und die gewünschte Datenbank angeben. Ein kleines Bash-Script macht das Leben einfacher:

```
#!/bin/bash
mysql -host=localhost -user=root -p DATENBANK
```

Nachdem wir lokal arbeiten, ist der Datenbankserver „localhost", der einzige bisher existierende User ist „root" und mit dem Parameter `-p` teilen wir dem Programm mit, dass es uns nach dem Passwort fragen soll. „DATENBANK" ist schließlich der Name der Datenbank, mit dem man arbeiten will (z. B. test). Nach dem Aufruf erscheint beispielsweise:

```
Enter password:
Enter password: *********
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 24 to server version: 4.04.24

Type 'help' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

In der letzten Zeile steht der leere Prompt. Hier werden Befehle eingegeben, mit denen Sie Ihre Daten bearbeiten können. Ohne etwas SQL-Kenntnisse kommen Sie nicht weiter. Hier ist gleich der erste Befehl: `quit` beendet den MySQL-Client wieder. Und verinnerlichen Sie schon jetzt die wichtigste Regel: Alle SQL-Befehle werden mit einem Strichpunkt abgeschlossen! Die Kommandozeile von `mysql` erinnert übrigens stark an die Bash, Sie können mit den Cursortasten navigieren und so Kommandos korrigieren und wieder abrufen.

Für tiefer gehende Informationen rund um MySQL müssen wir leider auf die im Anhang aufgeführte Literatur verweisen. Aber Sie sollen nicht ohne einen Teaser bleiben! Wir zeigen einfach einmal anhand eines Listings, wie eine Tabelle aussehen kann, wie man Daten einträgt und wie Abfragen aussehen können. Die

Tabelle haben wir mit dem grafisch orientierten MySQL-Admin „zusammengeklickt“. (Das Programm generiert dann eine SQL-Anweisung der Art „CREATE TABLE . . . “ und schickt sie an den Datenbankserver.) Unser Listing beginnt mit dem Abruf von Informationen über die Tabelle(n) in der Datenbank „test“ und ihren Aufbau:

```
mysql> show tables;
+----------------+
| Tables_in_test |
+----------------+
| Telefon        |
+----------------+
1 row in set (0.00 sec)

mysql> desc Telefon;
+--------+-------------+------+-----+---------+----------------+
| Field  | Type        | Null | Key | Default | Extra          |
+--------+-------------+------+-----+---------+----------------+
| Id     | int(11)     |      | PRI | NULL    | auto_increment |
| Nummer | varchar(10) |      |     |         |                |
| Name   | varchar(40) |      |     |         |                |
+--------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql> insert into Telefon (Name,Nummer)
    -> values('Michael Kofler','112578');
Query OK, 1 row affected (0.03 sec)

mysql> insert into Telefon (Name,Nummer)
    -> values('Juergen Plate','112579');
Query OK, 1 row affected (0.01 sec)

mysql> insert into Telefon (Name,Nummer)
    -> values('Hannibal Lecter','112581');
Query OK, 1 row affected (0.00 sec)

mysql> select * from Telefon;
+----+--------+-----------------+
| Id | Nummer | Name            |
+----+--------+-----------------+
|  1 | 112578 | Michael Kofler  |
|  2 | 112579 | Juergen Plate   |
|  3 | 112581 | Hannibal Lecter |
+----+--------+-----------------+
3 rows in set (0.03 sec)

mysql> select * from Telefon where Name='Michael Kofler';
+----+--------+-----------------+
| Id | Nummer | Name            |
```

```
+----+--------+-----------------+
|  1 | 112578 | Michael Kofler  |
+----+--------+-----------------+
1 row in set (0.00 sec)

mysql> select * from Telefon where Name like 'Mic%';
+----+--------+-----------------+
| Id | Nummer | Name            |
+----+--------+-----------------+
|  1 | 112578 | Michael Kofler  |
+----+--------+-----------------+
1 row in set (0.00 sec)

mysql> quit
```

Wie Sie sehen, geht das Eintragen und Abfragen recht einfach und da das Kommando `mysql` wie jedes Linux-Kommando Ein- und Ausgabe-Umleitung verträgt, hätte man den ganzen Dialog auch per Shell-Script erledigen können. Wie schon erwähnt, gibt es auch Programmierschnittstellen für viele Sprachen – und SQL bleibt immer gleich. Für Abfragen gibt es natürlich noch viele andere Möglichkeiten (das Prozentzeichen bei der letzten Abfrage wirkt beispielsweise als Jokerzeichen); sei es das Herausschneiden einzelner Felder oder Sortieren nach diversen Kriterien.

Die Homepage von MySQL ist *http://www.mysql.com/*; es gibt sogar eine spezielle deutsche Homepage unter *http://www.mysql.de/*.

Kapitel 10

Programmierung

Einführung

Ein Linux-Buch für Studenten ohne ein Kapitel über Programmierung wäre unvollständig. Nur stellt sich hier den Autoren die Frage, was man eigentlich schreiben soll. Die Informatikstudentinnen bekommen eine fundierte Programmierausbildung (meist C, C++ und Java), ebenso die Studenten der technischen Fachbereiche. Für diese Gruppe wäre dieses Kapitel eigentlich sinnlos. Andererseits kommen Geisteswissenschaftler und Wirtschaftler meist selten in die Verlegenheit, programmieren zu müssen. Es soll sogar Studenten geben, die das Ausfüllen einer Excel-Tabelle schon für eine Programmiertätigkeit halten. Andererseits können schon einfache Programmierkenntnisse das Leben leichter machen – was wir beim Kapitel über die Shell anklingen ließen.

Wenn Sie mit gängigen Distributionen arbeiten, installieren Sie zumeist nur fertig kompilierte Programme (so genannte Binärpakete). Es kann allerdings auch vorkommen, dass Sie keine vorkompilierte Version des Programms zum Download finden. In solchen Fällen müssen Sie den Quellcode herunterladen und das Programm selbst kompilieren. Deshalb geben wir Ihnen dazu einige einführende Tipps.

Danach stellen wir kurz einige Entwicklungsumgebungen vor, die sich für kleinere und größere Projekte eignen (das iat dann auch was für die Informatikerinnen) und von der Programmiersprache unabhängig sind. Sie stützen sich meist auf die Kommandozeilenversion der jeweiligen Compiler und Debugger. Zum Schluss gibt es dann einen knappen Kurs der Sprache *Perl*, eine Interpretersprache, die sich für administrative Aufgaben genauso eignet wie für Webanwendungen, und für die es vorgefertigte Module für nahezu jeden Zweck gibt. Auch ist der Umstieg von Perl zu PHP recht leicht.

10.1 Bibliotheken

Praktisch alle Linux-Programme verwenden dieselben Standardfunktionen, beispielsweise zum Zugriff auf Dateien, zur Ausgabe am Bildschirm, zur Unterstützung von X etc. Es wäre sinnlos, wenn jedes noch so kleine Programm all diese Funktionen unmittelbar im Code enthalten würde – riesige Programmdateien wären die Folge. Stattdessen bauen die meisten Linux-Programme auf so genannten *shared libraries* auf: Bei der Ausführung eines Programms werden automatisch auch die erforderlichen Libraries (Bibliotheken) geladen. Der Vorteil: Wenn mehrere Programme Funktionen derselben Library nutzen, muss diese nur einmal geladen werden.

Bibliotheken spielen eine zentrale Rolle dabei, ob und welche Programme auf Ihrem Rechner ausgeführt werden können. Fehlt auch nur eine einzige Bibliothek (bzw. steht sie in einer zu alten Version zur Verfügung), kommt es sofort beim Programmstart zu einer Fehlermeldung. Damit Sie in solchen Fällen nicht ganz hilflos den Tiefen der Linux-Interna ausgeliefert sind, vermittelt dieser Abschnitt einige Grundlageninformationen zu Bibliotheken.

Im Laufe der Linux-Geschichte hat es mehrmals ebenso grundlegende wie inkompatible Veränderungen der Linux-Bibliotheken gegeben, beispielsweise die Umstellung vom so genannten a.out-Format auf das ELF-Format oder die Umstellung von der Bibliothek libc 5 auf die glibc 2, die auch mit dem Namen libc 6 angesprochen wird.

Das Problem bei den Bibliotheksumstellungen besteht darin, dass kompilierte Programme nur dann ausgeführt werden können, wenn die dazu passenden Bibliotheken installiert sind und auch gefunden werden können.

Mit dem Kommando `ldd` kann man feststellen, welche Bibliotheken von einem Programm benötigt werden. `ldd` wird als Parameter der vollständige Dateiname des Programms übergeben. Als Reaktion listet `ldd` alle Libraries auf, die das Programm benötigt. Außerdem wird angegeben, wo sich eine passende Library befindet und welche Libraries fehlen bzw. nur in einer veralteten Version zur Verfügung stehen. Wenn `ldd` das Ergebnis *not a dynamic executable* liefert, handelt es sich um ein Programm, das alle erforderlichen Bibliotheken bereits enthält (ein statisch gelinktes Programm).

Kurz einige Informationen zur Nomenklatur der Libraries: Das Kürzel `.so` weist darauf hin, dass es sich um eine *shared library* handelt (im Gegensatz zu `.a` für statische Libraries). Die folgende Ziffer gibt die Hauptversionsnummer an. In den typischen Library-Verzeichnissen (zumeist `/lib`, `/usr/lib`, `/usr/local/lib`, `/usr/X11R6/lib` und `/opt/lib`) befinden sich oft Links von der Library-Hauptversion auf die tatsächlich installierte Library.

Beim Start eines Programms ist der so genannte *runtime linker* `ld.so` dafür zuständig, alle Bibliotheken zu finden und zu laden. `ld.so` berücksichtigt dabei alle in der Umgebungsvariablen `LD_LIBRARY_PATH` enthaltenen Verzeichnisse. Außerdem wertet der Linker die Datei `/etc/ld.so.cache` aus. Dabei handelt

es sich um eine Binärdatei mit allen relevanten Bibliotheksdaten (Versionsnummern, Zugriffspfade etc.). Das Kommando `ldconfig` wertet `/etc/ld.so.conf` aus. Diese Datei enthält eine Liste mit allen Bibliotheksverzeichnissen. (Die Verzeichnisse `/lib` und `/usr/lib` werden auf jeden Fall berücksichtigt und fehlen daher meistens in `ld.so.conf`.) `ldconfig` muss ausgeführt werden, wenn neue Bibliotheken manuell installiert werden! (Andernfalls sind die Bibliotheken für das System nicht sichtbar.)

10.2 Programme selbst kompilieren

Auch für Nicht-Programmierer gibt es manchmal Gründe dafür, Linux-Programme selbst zu kompilieren: Entweder finden Sie für das gewünschte Programm und Ihre Distribution kein fertiges Binärpaket oder Sie möchten das Programm mit einer vom Standard abweichenden Konfiguration kompilieren. Zuvor müssen jedoch einige Voraussetzungen erfüllt sein:

- Die *GNU Compiler Collection* (Pakete `gcc` und `gcc-c++`) muss installiert sein. Diese Pakete enthalten Compiler für C und C++.
- Hilfswerkzeuge wie `make`, `automake`, `autoconf` etc. müssen installiert sein. Diese Programme sind für die Konfiguration und Durchführung des Kompilationsprozesses erforderlich.
- Die Entwickler-Versionen diverser Bibliotheken müssen installiert sein (Pakete `glibc-devel` und `libxxx-devel`).

Im Internet finden Sie den Quellcode zumeist in komprimierten TAR-Archiven (Kennung `*.tar.gz` oder `*.tgz` oder `*.tar.bz2`). Nach dem Download entpacken Sie den Code in ein lokales Verzeichnis:

```
user$ tar xzf name.tar.gz     (für .gz oder .tgz)
user$ tar xjf name.tar.bz2    (für .bz2)
user$ cd name
```

Zum Verpacken von Quellcode ebenfalls beliebt sind SRMP-Pakete mit der Dateikennung `*.src.rpm`. Das ist eine Variante zu gewöhnlichen RPM-Paketen, aber eben für den *source code*. Zur Installation führen Sie wie üblich `rpm -i name.src.rpm` aus (siehe auch Seite 590). Es hängt von der Distribution ab, wo der Quellcode nun tatsächlich landet, in der Regel irgendwo unter `/usr/src`.

Ausgehend von diesem Verzeichnis finden Sie üblicherweise für jedes Quellcodepaket mehrere Dateien. `SOURCES/name.tar.xxx` enthält den eigentlichen Code als TAR-Archiv. `SOURCES/name-xxx.patch` oder `SOURCES/name.dif` enthält distributionsspezifische Veränderungen am ursprünglichen Code. Falls Sie die Codedateien entsprechend ändern (patchen) möchten, führen Sie das Kommando `patch < name.dif/patch` aus. `SPECS/name.spec` enthält eine Paketbeschreibung, die auch für die Erstellung von RPM-Paketen dient.

Zum Kompilieren und Installieren von Programmen sind drei Kommandos erforderlich: `.configure`, `make` und `make install`. Die drei Kommandos werden im Folgenden näher beschrieben. Dabei wird vorausgesetzt, dass Sie sich im Quellcodeverzeichnis befinden.

`configure` ist ein Script, das testet, ob alle erforderlichen Programme und Bibliotheken verfügbar sind. Nach dieser Systemanalyse adaptiert `configure` die Datei `Makefile`, die alle Kommandos enthält, um die diversen Codedateien zu kompilieren und zu linken. Bei manchen (zumeist eher kleineren Programmen) kann es sein, dass es das Script `configure` nicht gibt. In diesem Fall führen Sie sofort `make` aus.

`make` löst die Verarbeitung der Compile- und Link-Kommandos aus. Sie sehen nun (manchmal schier endlose) Nachrichten und Warnungen der verschiedenen Compiler-Läufe über das Konsolenfenster huschen. Solange kein Fehler auftritt, können Sie diese Meldungen getrost ignorieren. Als Ergebnis sollte sich im Quellcodeverzeichnis nun die ausführbare Datei *name* befinden. In vielen Fällen können Sie das Programm nun sofort starten (Kommando `./name`) und testen (sofern nicht eine spezielle Konfiguration erforderlich ist oder das Programm nur durch Init-V-Scripts korrekt gestartet werden kann).

Der letzte Schritt besteht darin, das Programm allen Benutzern zugänglich zu machen. Dazu müssen die Programm- und eventuell auch Bibliotheksdateien in öffentlich zugängliche Verzeichnisse kopiert werden. Das erfordert *root*-Rechte. Vor der Ausführung von `make install` sollten Sie sicherstellen, dass das betreffende Programm nicht schon installiert ist! Wenn das der Fall ist, sollte es vorher deinstalliert werden.

Während des Kompilierens können vielfältige Probleme auftreten. Am wahrscheinlichsten ist, dass irgendwelche Compiler-Hilfswerkzeuge oder zum Kompilieren notwendige Bibliotheken (die Entwickler-Versionen dieser Bibliotheken fehlen). Diese Probleme werden in der Regel bereits durch `configure` festgestellt und lassen sich meist relativ leicht beheben, indem das fehlende Paket einfach installiert wird. Schon schwieriger wird es, wenn `configure` nach Bibliotheken verlangt, die in Ihrer Distribution nicht oder nicht in der erforderlichen Version verfügbar sind: Dann müssen Sie sich im Internet auf die Suche nach der betreffenden Bibliothek machen und eventuell zuerst diese kompilieren.

Wesentlich problematischer ist es, wenn während der Kompilierung ein Syntaxfehler auftritt, die Kompilation also mit einer Fehlermeldung abbricht. Schuld daran ist oft nicht ein Programmfehler, sondern eine Inkompatibilität zwischen Ihrem Compiler und dem Code. Manche Programme können nur mit einer bestimmten Version von `gcc` kompiliert werden (oft *nicht* mit der gerade neuesten `gcc`-Version!). Die Lösung besteht hier darin, die gewünschte Compiler-Version zu installieren. Auch hierzu finden Sie im Internet oder in den `README`-Dateien zum Quellcode oft genaue Anweisungen.

Der Platz in diesem Buch reicht nicht aus, um auch eine Einführung in die Programmierung in C und C++ zu geben. Da wir aber in der Vergangenheit immer wieder diesbezügliche Fragen erhalten haben, finden Sie hier eine kurze

Anleitung, wie Sie das klassische „Hello World“-Programm in C und C++ programmieren und kompilieren. Für die C-Version schreiben Sie mit einem Editor die folgenden Zeilen in die Datei `hello.c`:

```
#include <stdio.h>
int main(void)
  {
  printf("Hello World!\n");
  return 0;
  }
```

Mit den folgenden Kommandos kompilieren Sie das Programm und führen es aus (die Option `-Wall` schaltet alle Warnungen des Compilers ein und die Option `-o hello` benennt die fertige Binärdatei):

```
user$  gcc -Wall -o hello hello.c
user$  ./hello
Hello World!
```

Der vergleichbare Code in C++ sieht so aus:

```
#include <iostream>
int main()
  {
  std::cout << "Hello World!\n";
  return 0;
  }
```

Zum Kompilieren verwenden Sie nun `g++` statt `gcc`:

```
user$  g++ -Wall -o hello hello.cpp
user$  ./hello
Hello World!
```

10.3 Entwicklungsumgebungen

Eine Entwicklungsumgebung, genauer eine integrierte Entwicklungsumgebung, Integrated Development Environment (IDE), ist eine grafische Bedienoberfläche, die alle für die Programmentwicklung notwendigen Werkzeuge wie Editor, Compiler, Debugger etc. zusammenfasst. Bei einigen Umgebungen kommt noch eine komplette Projektverwaltung hinzu, die sich oft auch für Teams eignet. Kern ist meist der Editor mit Syntaxhervorhebung. Oft stützt sich die IDE auf die Kommandozeilen-Programme, wie wir sie im vorhergehenden Abschnitt verwendet haben. Der im vorhergehenden Kapitel auf Seite 452 besprochene Editor SciTE bietet da schon Ansätze, denn man könnte ihm beibringen, den Compiler `gcc` mit dem gerade bearbeiteten Text aufzurufen. Er bietet auch ein Statusfenster, in dem man die Meldungen des Compilers sehen könnte.

Eine IDE bietet natürlich mehr. Neben den oben erwähnten Tools kann das Programm auch gleich ausgeführt und seine Ausgabe festgehalten werden. Durch geeignete Compiler-Optionen ist auch Debugging mit Quelltextbezug möglich. Die Projektverwaltung generiert nebenher auch die für `make` und `configure` nötigen Dateien – gegebenenfalls auch Headerfiles und die Grundlagen für die Dokumentation. In diesem Abschnitt stellen wir einige IDEs vor. Da der Ablauf bei allen IDEs relativ ähnlich ist, gehen wir nur bei einer, `kdevelop`, genauer auf die Abläufe bei der Anwendung ein.

10.3.1 xwpe

Vor langer, langer Zeit (um das Jahr 1992) entwickelte Fred Kruse einen textbasierten Editor mit Dateibrowser, der wie die damaligen DOS-Tools aussah. (Erinnert sich noch jemand an Turbo C 2.0 für DOS?) Immerhin war er bunt, klein und bedienerfreundlich. Viele Programmier schätzten gerade diese Eigenschaften. Dieser Editor hieß *Window Editor (we)*. Spätere Versionen ermöglichten dann auch das Compilieren, Ausführen und Debuggen von Programmen aus dem Editor heraus. Nun hieß das Programm *Window Programming Eenvironment (wpe)* und es lief entweder auf der Konsole oder in einem X-Terminalfenster. Die Version, die unter X läuft, ist unter dem Namen *xwpe* verfügbar. Sie hat aber die tastaturgesteuerte Bedienung des Originals beibehalten. Wer damit spielen will, findet die aktuelle Version bei Identical Software unter *http://www.identicalsoftware.com/xwpe/*.

10.3.2 C-Forge

Etwas aus dem Rahmen fällt auch die „C-Schmiede“, denn es handelt sich um eine kommerzielle, grafische IDE. C-Forge eignet sich, wie der Name schon andeutet, für die Sprachen C und C++. Es gibt diverse Versionen, darunter auch für Linux. Wir erwähnen das Programm nur deshalb, weil es für Linux auch eine kostenlose Testversion mit eingeschränkter Funktionalität gibt. Diese ist aber immer noch für kleine Vorhaben brauchbar.

C-Forge erzeugt für ein Projekt ein passendes Makefile und unterstützt sogar eine Versionskontrolle mit CVS. So lassen sich die Projekte schnell und unkompliziert warten.

Die kostenlose Version von C-Forge erhalten Sie im Internet unter *http://www.codeforge.com*.

10.3.3 Anjuta

Anjuta ist eine integrierte Entwicklungsumgebung für Gnome. Das Projekt wurde von Naba Kumar initiiert und ist noch im Entwicklungsstadium (Version 2.x ist im Moment aktuell), kann aber schon als Alternative zum KDE-Projekt

KDevelop (siehe unten) angesehen werden. Anjuta bietet unter anderem folgende Features:

- Projektmanagement und Buildmanagement für C und C++
- Java, Perl, Pascal und andere Sprachen werden ohne Projektmanagment unterstützt.
- Druiden zur Genererierung von GTK- und GNOME-Applikationen
- Funktionen, Klassen, Structuren etc. können über einen Symbolbrowser schnell eingesehen werden.
- Wizard zur Projektgenerierung
- automatische Erstellung von Makefiles
- automatische Code-Vervollständigung, Syntaxhervorhebung und Formatierung (auch für andere Sprachen)
- integrierter, auf dem GNU-Debugger basierender Debugger
- Unterstützung von Plug-Ins

Wenn es kein Komplettpaket für Ihre Distribution gibt, kann die Installation manchmal ein paar kleine Probleme bringen: Sie müssen gegebenenfalls ein paar notwendige Bibliotheken nachladen. Zu finden ist Anjuta unter *http://www.anjuta.org/*. Im Unterverzeichnis `documentations/subpage/documents/de/anjuta-tutorial/` finden Sie auch ein deutschsprachiges Tutorial.

10.3.4 KDevelop

KDevelop ist die integrierte Entwicklungsumgebung des KDE-Projekts. KDevelop bietet Funktionen wie Code Completion, Assistenten für das Erstellen neuer Projekte, Klassen oder auch nur einzelner Dateien. Gleichzeitig stellt es Frontends für den Debugger `gdb`, den Debugger und Profiler Valgrind, die GNU Compiler Collection, das Dokumentationssystem Doxygen sowie Versionskontrollsysteme wie CVS und Subversion zur Verfügung. Vieles, was wir hier schreiben, trifft auch für andere Entwicklungsumgebungen wie Anjuta zu.

Die KDevelop-IDE ist öffentlich unter der GPL erhältlich und unterstützt u. a. Qt-, KDE-, GNOME-, C++- und C-Projekte. Daneben arbeitet es auch mit einer Vielzahl von Programmiersprachen zusammen und lässt sich durch Plug-Ins erweitern. Zusätzlich können weitere externe Programme in die Menüs eingebunden werden.

KDevelop eignent sich zum schnellen Erstellen von Programmen für die Konsole, aber auch für die grafische Oberfläche X. Dem Programmierer werden die Erstellung und Verwaltung aller benötigten Projektdateien abgenommen. Es entsteht

so ein fertiges Projekt, das alle wichtigen Dateien (Quellcode, configure-Script, Makefile und Dokumentation) enthält. Dieses Projekt kann anschließend in ein tar-Archiv gepackt und veröffentlicht werden. KDevelop eignet sich durch seinen Funktionsumfang und seine Erweiterbarkeit auch für große Projekte, macht es aber Anfängern schwer, einen Zugang zum gesamten Funktionsumfang der Entwicklungsumgebung zu finden.

Seit der Version 3.0 hat KDevelop eine völlig neue grafische Oberfläche namens IDEAl (siehe Abbildung) und beherrscht viele neue Programmiersprachen, darunter ADA, Shell, C, C-# (PlugIn), C++, Fortran, Haskell, Java, Objective-C, Pascal, Perl, PHP, Python, Ruby, SQL und andere.

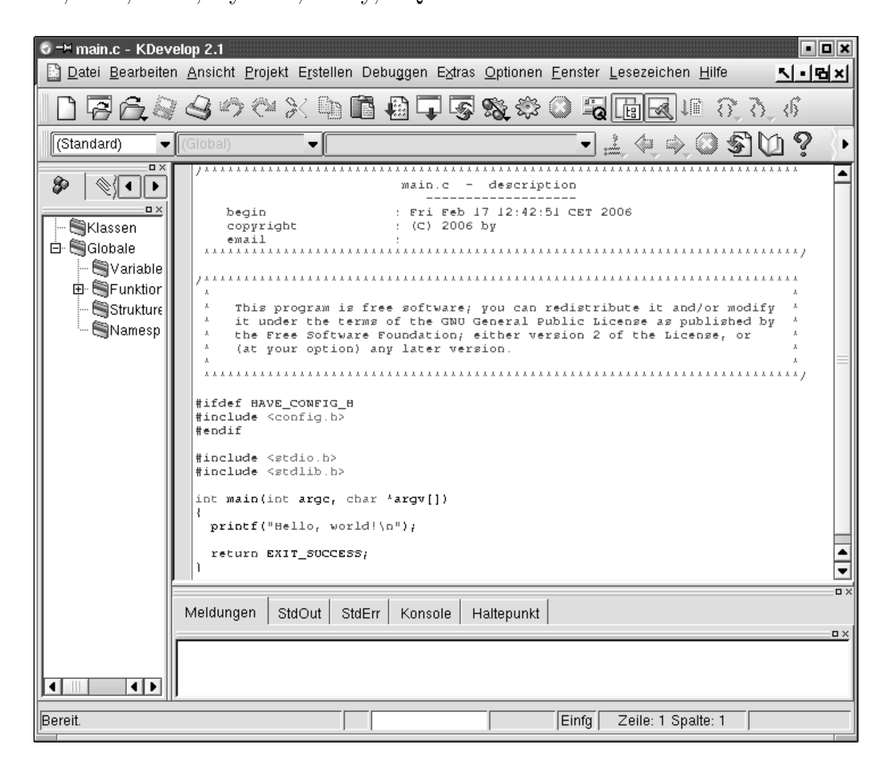


Abbildung 10.1: Der Bildschirm von KDevelop

Die Arbeit mit KDevelop beginnt mit dem Anlegen eines neuen Projekts. Sie gehen nach dem Start des Programms zum Menü *Projekt*. In diesem Menü wählen Sie *Neues Projekt*. Es wird ein Fenster angezeigt, in dem Sie aus der Liste die gewünschte Sprache, beispielsweise C, aussuchen. Bei dieser Sprache gibt es ein Untermenü mit verschiedenen Möglichkeiten, u. a. Erstellen einen Kernel-Moduls, grafischer Anwendungen und das „simple hello world“, das für eine textbasierte Anwendung geignet ist. Mit *Weiter* gelangen Sie zum nächsten Fenster, in dem Sie einen Namen für Ihr Projekt wählen können. Das war es eigentlich

schon. Sie drücken nun solange den Button *Weiter*, bis Sie mithilfe des Buttons *Abschließen* (bei der englischen Version *Create*) das Projekts erstellen.

KDevelop generiert anschließend eine ganze Reihe von Dateien. Wenn im Fenster *Prozesses* die Meldung „READY“ erscheint, sind alle Dateien komplett und durch Anklicken der Schaltfläche *Exit* kann der Application Wizzard beendet werden. Nach der Erzeugung einer C-Applikation im Application Wizzard kann mit dem Programmieren begonnen werden. Der Wizzard hat dazu bereits ein Rumpf-Programm *main.c* erstellt.

```
/*******************************************************************
                       main.c  -  description
                          -------------------
  begin                : Don Mar  29 10:49:48 MET 2006
  copyright            : (C) 1999 by Hugo Habicht
  email                : habicht@somewhere.de
 ******************************************************************/

#ifdef HAVE_CONFIG_H
#include <config.h>
#endif

#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
  {
  printf("Hello, world!\n");
  return EXIT_SUCCESS;
  }
```

Dieses Programm können Sie gleich als Basis für Ihre Applikation verwenden, indem Sie die `printf`-Zeile durch Ihr Programm ersetzen. Es ist auch sinnvoll, das Hello-World-Programm gleich einmal zu übersetzen und zu starten, um zu sehen, ob alle notwendigen Dateien angelegt wurden. Außerdem werden beim ersten Übersetzungs-Durchlauf auch noch weitere Dateien angelegt. Danach löschen Sie die Zeile `printf("Hello, world!\n");` und tippen Ihr Programm ein.

Die wichtigsten Funktionen von *KDevelop* lassen sich über Icons in der Menüleiste steuern. Dazu gehören unter vielen anderen:

- **Projekt öffnen:** Mit diesem Knopf können Sie ein gespeichertes Projekt öffnen und daran weiterarbeiten.
- **Datei öffnen:** Öffnet einen (weiteren) Quelltext
- **Datei speichern:** Abspeichern aller Quelltexte (wird beim Compilieren automatisch gemacht)

- **Quelltext compilieren:** Nur der aktuell im Editor sichtbare Quelltext wird übersetzt. Dabei wird kein ausführbares Programm erzeugt.
- **Alle Quelltexte compilieren:** (Make) Alle Quellen werden neu übersetzt und alle Objekt-Dateien zu einem Programm zusammengelinkt. Hierbei entsteht ein ausführbares Programm.
- **Komplettes Projekt neu compilieren:** (Build) Es wird ein ausführbares Programm erstellt. Alle Quelltexte werden neu übersetzt und die Objektmodule dann zu einem Programm zusammengelinkt und dieses anschließend gestartet.
- **Debuggen:** Fehlersuche im Binärprogramm mittels `gdb`.
- **Run:** Übersetztes Programm ausführen. Öffnet ein Shell-Fenster und startet darin das erstellte Programm

10.3.5 Komodo von ActiveState

Komodo ist eine kommerzielle IDE für Scriptsprachen, die das Erstellen von Scripts und die Fehlersuche darin erleichtert. Für den privaten Gebrauch gibt es auch eien kostenlose Version. Interessant ist auch, dass Komodo für Linux, Windows und Mac erhältlich ist (übrigens stammt auch der bekannteste Perl-Interpreter für Windows von ActiveState). Komodo unterstützt in der aktuellen Version 3.5 die Sprachen Perl, PHP, Python, Ruby, XSLT und Tcl. Die zahlreichen Features wie Realtime-Debugging oder das Regular-Expression-Toolkit sind beeindruckend.

Gerade wenn Sie mit einer Scriptsprache noch nicht so recht warm geworden sind, leistet die IDE über einen Echtzeit-Debugger gute Dienste. Fehler aller Arten werden schon beim Eintippen des Programms erkannt. Scriptsprachen sind normalerweise für kleinere Programme gedacht (80 % aller Scripts liegen bei etwa 150 Zeilen). Diese lassen sich wunderbar in Echtzeit debuggen. Nimmt das Programm ein größeres Ausmaß an, wird Komodo langsamer.

Ein echtes Feature ist der Assistent für reguläre Ausdrücke, das RegEx-Toolkit. Komplexere Ausdrücke lassen einen recht schnell den Überblick verlieren und im Programmzusammenhang auch nur aufwändig testen. Hier hilft der Assistent bei der Entwicklung und man übernimmt einen getesteten Ausdruck ins Programm.

Besonders erfreulich sind die vielen Detaillösungen. Um etwa bei größeren Arbeiten auch die Übersicht zu behalten, können Sie alles, was in geschweiften Klammern steht, für ein effizienteres Arbeiten „einfalten“. Die Übersichtlichkeit steigt sprunghaft an. Im Paket enthalten ist eine sehr ausführliche Hilfe, in der alle Funktionen übersichtlich gegliedert sind und die über den integrierten Mozilla-Browser ausgeliefert wird.

Kostenlos wird immer eine etwas ältere Version geliefert, die aktuelle Version ist für etwas 300 Dollar zu haben. Weitere Informationen finden Sie unter *http://www.activestate.com*.

10.3.6 Geany

Wer zur Arbeit an kleinen Programmen nicht gleich mit teamfähigen Boliden wie KDevelop und anderen oben beschriebenen Umgebungen gewissermaßen mit Kanonen auf Spatzen schiessen will, findet vielleicht in Geany das Richtige. Es wird unter der GPL vertrieben. Geany ist ein noch recht junges Projekt, das sich zum Ziel gesetzt hat, eine kleine und schnelle IDE zu bieten. Gleizeitig wird eine möglichst große Unabhängigkeit von einer bestimmten Desktop-Umgebung angestrebt. So stützt es sich lediglich auf die GTK2-Bibliothek. Die wichtigsten Features sind:

- Syntax-Highlighting für alle gängigen Programmiersprachen (u. a. C, C++, Java, Perl, PHP, Pyton, Bash, Pascal und sogar HTML)
- Code-Vervollständigung, auch automatische Vervollständigung von Strukturen wie „if“, „for“ oder „while“ sowie von XML- and HTML-Tags
- Call-Tipps
- Referenztabellen zur schnellen Navigation im Quelltext

In der Programmkonfiguration sind bereits die wichtigsten Compiler eingetragen. Die o. g. Referenztabellen erlauben den direkten Sprung zu einer jeden Prozedur/Funktion oder Objektdefinition im Quelltext. Die Tabelle ist übersichtlich als Baum strukturiert.

Die Meldungen beim Übersetzen und Linken werden wie bei den anderen IDEs in einem extra Statusfenster angezeigt. Dort gibt es neben den drei Statusfenstern für die verschiedenen Meldungen noch die Möglichkeiten, Notizen oder Programmstücke zwischenzuspeichern, und ein Terminalfenster für direkte Shell-Eingaben.

Geany verwendet, wie gesagt, GTK (ab Version 2.6) und, falls Sie das Paket selbst übersetzen wollen, noch einen C/C++-Compiler sowie `make`. Geany lässt sich problemlos unter Debian (ab 3.1), Fedora Core 4, LinuxFromScratch und FreeBSD 6.0 installieren (mit heftigem Gefummel im Makefile soll es sogar unter Windows laufen). Fertige Binärpakete für Red Hat/Fedora finden Sie unter *http://geany.uvena.de/* und Debian-Pakete sind unter *http://debian.uvena.de/* zu finden. Debian- und Ubuntu-Benutzer können auch einfach die beiden folgenden Zeilen in der Datei `/etc/apt/sources.list` einfügen, um das Paket mittels `apt-get` zu installieren.

```
deb http://debian.uvena.de/ ./stable/
deb http://debian.uvena.de/ ./testing/
```

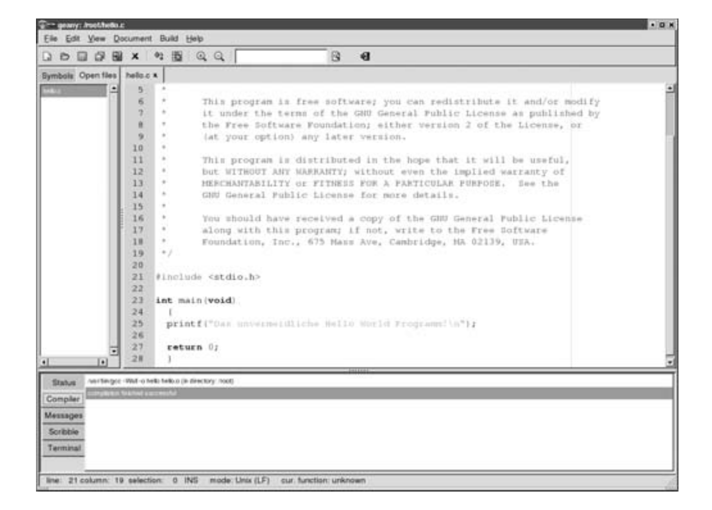


Abbildung 10.2: Geany präsentiert sich wie die großen IDEs.

Nach dem Eintrag das Kommando `apt-get update` nicht vergessen! Für die Suse-Distribution gibt es noch nichts Fertiges.

10.3.7 Eclipse

Eclipse ist eine universell einsetzbare Entwicklungsumgebung mit Projektverwaltung, Debugger und CVS-Unterstützung speziell für Java. Mit passenden Plug-Ins lässt sich diese IDE auch auf andere Sprachen erweitern. Dem Programmierer stehen automatische Syntax-Highlighting, Syntaxergänzung, Online-Hilfe und ein Template-Editor zur Verfügung. Syntaxergänzung und Templates arbeiten Hand in Hand.

Bei der automatischen Code-Ergänzung berücksichtigt Eclipse auch eigene Funktionen, Klassen, Methoden und Variablen. Zu Funktionen und Klassen gehörende Kommentare zeigt die Auto-Ergänzung übrigens auch in ihrer Auswahlliste, sobald Sie einen Eintrag aus der Liste anklicken. Auf Wunsch durchsucht die IDE auch das komplette Projekt nach Zeichenketten. Als Fundstellen zeigt Eclipse in einem Ergebnisfenster zunächst nur die Dateinamen. Ein Doppelklick auf einen Namen führt dann direkt zur Fundstelle in der jeweiligen Datei.

Der Debugger akzeptiert Breakpoints und zeigt die Werte von Variablen zur Laufzeit an. Die Programmierumgebung ist in so genannte Perspectives aufgeteilt, was man mit „Sichtweisen" übersetzen könnte. Diese bestehen aus unterschiedlichen Unterfenstern, die bedarfsweise eingeblendet werden. Wer beispielsweise gerade bei der Fehlersuche ist, schaltet die Debug-Perspective ein.

Die Bedienung von Eclipse ist einfach und übersichtlich. Funktionen und Methoden lassen sich „einfalten“ und damit aus dem Blickfeld schaffen. Eine Formatierungsfunktion hilft den Tipp-Chaoten, einen lesbaren Quelltext zu erstellen. Einziger Wehrmutstropfen: Die Installation ist etwas kniffelig.

Speziell für Java-Programmierer bietet diese kostenlose IDE Profi-Power zum Nulltarif. Zu finden ist Eclipse unter *http://www.eclipse.org/*.

10.3.8 Gambas

Für viele Umsteiger auf Linux war der Verlust einer einfachen Programmiersprache wie Basic schwer zu verschmerzen. Gambas steht für „Gambas Almost Means BASic“ und stellt ein Programmiersystem bereit, das Visual Basic ähnelt. Mit Gambas können Sie grafische Oberflächen mit wenigen Mausklicks erstellen und den Quelltext in einem komfortablen Editor schreiben. Da sich die Oberflächen von Gambas und Visual Basic ähneln, finden sich Umsteiger schnell zurecht. Jedoch weicht die Gambas-Syntax von Visual Basic ab, so dass sich VB-Programme nicht ohne Änderungen übernehmen lassen. Einfache Projekte ohne ActiveX-Steuerelemente kann man jedoch mit überschaubarem Aufwand anpassen.

Die Gambas-IDE enthält einen Debugger zur Fehlersuche, einen grafischen Formular-Editor und einen Menü-Editor – man kann sich also auch hier wie bei VB sein Programm „zusammenklicken“. Durch sein modulares Konzept bindet es Entwickler nicht an ein bestimmtes Toolkit, Gambas-Programme lassen sich auf Knopfdruck mit Qt- oder Gtk-Oberflächen versehen. Ähnlich wie beim Grafikprogramm Gimp verteilen sich recht schnell diverse Fenster auf dem Desktop.

Wählen Sie im Startbildschirm Neues Projekt, so führt ein Assistent Sie durch die einleitenden Schritte. Zunächst entscheiden Sie, ob Sie ein grafisches oder ein textbasiertes Projekt anlegen oder Code aus einem bestehenden Projekt kopieren möchten. Danach geben Sie den Namen des Projektverzeichnisses ein. Beim Einstieg in die Gambas-Programmierung helfen mitgelieferte Beispiele.

Jedes grafische Programm braucht mindestens ein Fenster oder *Formular*, das Eingaben des Benutzers in Empfang nimmt und Ausgaben anzeigt. Um ein neu erstelltes, leeres Formular zu füllen, verwenden Sie die zahlreichen Steuerelemente aus der Werkzeugsammlung. Neben dem in Arbeit befindlichen Formular zeigt Gambas den Quelltext-Editor, das Projektfenster mit allen zum Projekt gehörenden Dateien, die Werkzeugleiste mit verfügbaren grafischen Elementen und das Eigenschaftsfenster, in dem Sie Aussehen und Verhalten der Programmoberfläche festlegen. Das sind u. a. der Name, unter dem Sie das Element im Programm ansprechen, die Position und Größe des Fensters, dann Beschriftung, Schriftart und -größe sowie Vorder- und Hintergrundfarben.

Menüs erstellen Sie mit dem Menü-Editor, der bis zu vier Ebenen tiefe Menüs erlaubt. Den einzelnen Menüpunkten können Sie dann noch Tastenkürzel und

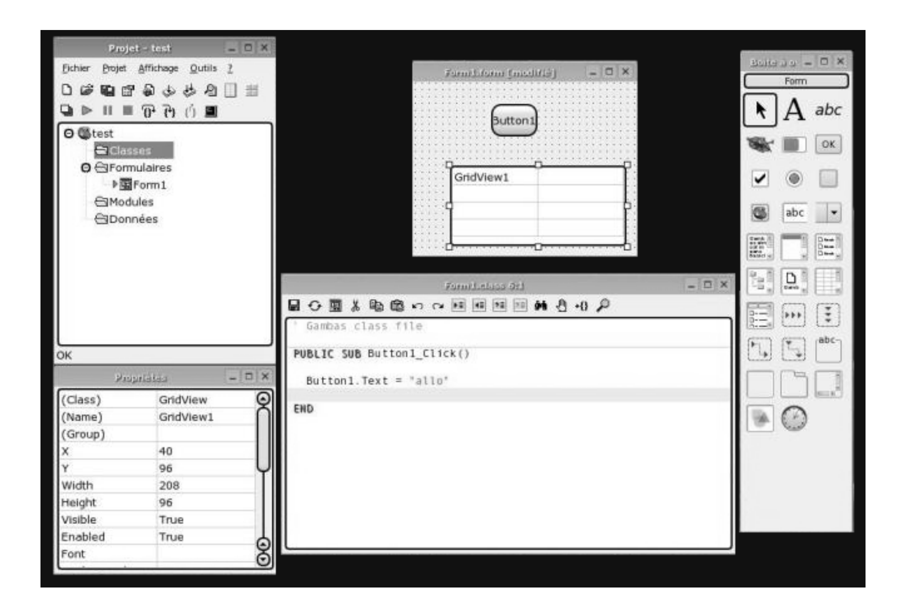


Abbildung 10.3: Gambas verteilt seine Fenster auf dem ganzen Desktop.

Symbole zuordnen. Ein Gambas-Programm lässt sich über Komponenten einfach um vorgegebene Funktionen erweitern. Zur Basisausstattung gehören Module für Datenbankzugriff und KDE-Widgets. Auch der Gambas-eigene Quelltext-Editor lässt sich komplett in eigene Programme einbinden. Daneben stehen einige Module zur Verfügung, die sich noch im Experimentierstadium befinden. Eine Stärke von Visual Basic ist der einfache Zugriff auf Datenbanken. Gambas unterstützt MySQL-, PostgreSQL- und SQLite-Datenbanken.

Fertige Programme lassen sich nicht nur zu einem ausführbaren Binary kompilieren oder zu einem Quelltextarchiv packen, sondern es gibt auch einen Assistenten, der automatisch wahlweise RPM- oder Debian-Pakete erzeugt.

Neben allem anderen verfügt Gambas über eine umfangreiche Hilfe in englischer Sprache. Sie finden dort eine Beschreibung der Schlüsselwörter und eine kurze Programmier-Einführung. Beim Umstieg hilft die Erklärung der Unterschiede zwischen Gambas und Visual Basic. Auch bei den einzelnen Befehlen weist die Hilfe immer wieder auf Unterschiede zu VB hin.

Gambas-Homepage: *http://gambas.sourceforge.net*

Neben Gambas gibt es noch weitere Basic-Projekte mit höchst unterschiedlichem Komfort. Darunter befinden sich:

XBasic: *http://www.maxreason.com/software/xbasic/xbasic.html*
HBasic: *http://hbasic.sourceforge.net*
WxBasic: *http://wxbasic.sourceforge.net*

KBasic: *http://www.kbasic.de*
PowerBasic: *http://www.powerbasic.com*
RealBasic: *http://www.application-systems.de/realbasic/*

10.4 Programmierung mit Perl

Ursprünglich konzipiert, um große Mengen Text schnell zu durchsuchen und zu analysieren, wird Perl mittlerweile für nahezu alle denkbaren Zwecke eingesetzt. Daher kommt auch der Name „Practical Extraction and Report Language". Die Sprache wurde von einem einzelnen Entwickler entworfen, Larry Wall. Perl ist Freeware und auf vielen Rechnerplattformen verfügbar, unter anderem Linux, DOS/Windows und Mac OS. Perl ist

- eine interpretierte Sprache,
- für Erschließung, Aufbereitung und Neuformatierung beliebiger Texte,
- kann `C-Programme` sowie `sed-`, `awk-` und `shell-Scripts` ersetzen,
- überwindet alle Beschränkungen dieser Sprachen
- und hat den Spitznamen *Pathologically Eclectic Rubbish Lister*.

Wir wollen Ihnen in diesem Abschnitt eine Einführung in Perl geben und beschränken uns auf die prozeduralen Aspekte der Sprache. Wenn Sie Spaß an Perl gefunden haben, können Sie sich bestimmt die objektorientieren Aspekte alleine aneignen. Auch fehlt der Platz, um auf die zahllosen Module einzugehen, die es inzwischen gibt. Bevor Sie also selbst programmieren, sehen Sie erst einmal bei *http://www.cpan.org* nach.

Am bekanntesten ist die Verwendung als Programmiersprache für CGI-Scripte im WWW sowie im Bereich der Linux-Systemadministration. Aber auch für viele andere Aufgaben hat sich Perl bewährt:

- Datenbankmanipulation
- Text- und Dokumentenverarbeitung
- Datenextraktion und -reduktion
- Netzwerk-Administration
- Filesystem-Manipulation
- sicherheitsorientiertes Programmieren
- WWW-basiertes Informationsmanagement

- objektorientierte und funktionale Programmierung
- ...

Perl wurde von Beginn an als Mischung zwischen Shell- und C-Programmierung entwickelt. Es gibt daher zahlreiche Anleihen aus anderen Sprachen sowie verschiedene Perl-Spezialitäten. Dabei ist Perl im Gegensatz zu Shell-Scripts oder C-Programmen jedoch äußerst portabel.

Eigentlich ist Perl eine einfache Sprache – man braucht nicht viele reservierte Worte (Elemente der Sprachsyntax) zu kennen, um ein Perl-Programm zu schreiben und man kann es ausführen wie ein Shell-Script. Die von Perl benutzten Typen und Strukturen sind einfach zu verstehen und zu benutzen. Es gibt keine künstlichen Beschränkungen für die zu verarbeitenden Daten; Strings und Arrays können beliebig wachsen (solange der Speicher reicht) und sie sind darauf ausgelegt, trotzdem gut zu skalieren. Obwohl Perl in vieler Hinsicht sehr einfachgehalten wurde, so ist es auch eine sehr reiche Sprache, bei der es sehr viel zu lernen gibt. Das ist der Preis dafür, dass die einfachen Probleme einfach und die komplexen Aufgaben überhaupt zu lösen sind. Trotzdem ist Perl keine Sprache für große Projekte, es ist im besten und positivsten Sinn eine „Hacker-Sprache“. Larry Wall hat dazu noch bemerkt, dass man in Perl die Lösung eines Problems auf vielen Wegen erreichen kann – und dass alle Wege richtig sind. Perl ist somit auch eine Sprache, die der Bewusstseinsverengung der Programmierer entgegenwirkt.

Aufgrund seiner offenen Struktur kann Perl sehr leicht erweitert werden. Man kann Perl in andere Sprachen und andere Sprachen in Perl einbetten und sogar objektorientierte Bibliotheken in Perl benutzen. Durch den Mechanismus zum Modulimport können externe Definitionen so genutzt werden als wären sie in Perl eingebaut. Es gibt zahllose Erweiterungen für Perl, darunter Datenbankanbindungen, Anbindungen an das grafische Toolkit Tk, mathematische Erweiterungen und vieles mehr.

Obwohl eigentlich eine Interpretersprache, ist Perl eher eine Kreuzung aus Interpreter und Compiler. Zur Laufzeit wird das Perl-Script vom Interpreter relativ schnell in ein internes, bytecodeähnliches Format umgesetzt. Dabei werden, wie bei jedem Compiler, Syntaxchecks und Optimierungen durchgeführt (syntaktische und zum Compilierungszeitpunkt erkennbaren semantische Fehler, Probleme beim Einbinden von Bibliotheken usw.). Nach dem erfolgreichen Durchlauf des Compilers wird der Zwischencode an den Perl-Interpreter zur Ausführung weitergereicht. Compiler und Interpreter sind effizient und der typische Übersetzungszyklus dauert meist nur Sekunden. Zusammen mit den vielen Möglichkeiten Perls, auf Fehler „weich“ zu reagieren, erlaubt diese kurze Zykluszeit Rapid-Prototyping in Perl.

Perl-Dateien haben in der Regel die Endung `.pl` oder `.cgi`. Die Ausführung wird entweder durch expliziten Aufruf des Perl-Interpreters gestartet (z. B. perl tuwas.pl) oder bei Linux durch eine spezielle Kommentarzeile zu Beginn der Quelldatei.

10.4.1 Hello World

Das unvermeidliche erste Programm:

```
#!/usr/bin/perl
#
# Das unvermeidliche erste Programm
#
print "Hello world.";           # Ausgabe eines Textes
```

Fast jedes Perl-Programm beginnt unter Linux mit der Zeile

```
#!/usr/bin/perl
```

obwohl die Zeile von System zu System unterschiedlich aussehen kann, sagt sie dem Computer, was er bei der Ausführung zu tun hat. In diesem Fall soll er das Programm durch den Perl-Interpreter schicken. Alles Übrige sind Perl-Anweisungen, welche mit einem Strichpunkt beendet werden müssen, wie die letzte Zeile oben.

Die `print`-Funktion gibt Information aus. Im obigen Fall gibt sie die Zeichenkette *Hello world.* aus.

10.4.2 Ein Programm ausführen

Schreiben Sie das Beispielprogramm mit einem Text-Editor und speichern Sie es ab. Danach muss es ausführbar gemacht werden, beispielsweise mit dem Linux-Kommando

`chmod u+x` *progname*

Um das Programm zu starten, können Sie eine der folgenden Möglichkeiten wählen:

`perl` *progname*
`./` *progname*

Vielleicht haben Sie Glück und das Programm wird ausgeführt. Bei der Ausführung eines Perl-Programmes (Perl-Script) wird der Text zuerst kompiliert (interpretiert). Dieser Schritt erzeugt auch die Fehlermeldungen. Anschließend wird es ausgeführt. Die eigentliche Laufzeit ist vergleichbar mit einem C-Programm. Je nach Länge des Programmes muss mit einer gewissen Kompilierzeit gerechnet werden.

Möglicherweise erscheinen aber Fehlermeldungen oder es geschieht überhaupt nichts. Man kann beim Aufruf den Perl-Interpreter veranlassen, Warnungen und weitere hilfreiche Meldungen auszugeben, bevor das Programm ausgeführt wird.

10.4.3 Aufruf

Beim Aufruf des Perl-Interpreters (Linux-Version) lassen sich einige Optionen angeben (Aufruf-Beispiel: `perl -c test.pl` oder `perl -cwT test.pl`).

Optionen des Perl-Interpreters

Option	Bedeutung
-c	Das Perl-Script wird nicht ausgeführt, sondern nur auf syntaktische Richtigkeit überprüftö.
-d	Das Perl-Script wird im Debug-Modus ausgeführt.
-S	Das Script braucht nur als Dateiname ohne Pfad angegeben werden. Wenn es in einem der Verzeichnisse liegt, die in der PATH-Variablen des Betriebssystems genannt sind, findet Perl das Script auch ohne Pfadangabe.
-T	Das Script wird während der Ausführung streng auf Fehlerfreiheit überprüft.
-v	Gibt die Version des Perl-Interpreters aus
-w	Bewirkt, dass der Perl-Interpreter nicht nur Fehlermeldungen bei Syntaxfehlern ausgibt, sondern auch Warnungen bei möglichen Fehlern und logisch zweifelhaften Anweisungsfolgen.

10.4.4 Sprachkontext

Perl setzt keine Vereinbarung von Typen, Datenstrukturen, Variablen, Funktionen wie in anderen Programmiersprachen voraus. Für komplexe Probleme ist es jedoch sinnvoll, Variablen am Programmanfang zu definieren, vorzubesetzen und zu kommentieren. So werden viele Fehler im Vorfeld vermieden oder zumindest schneller gefunden. Wie leicht vertippt man sich an irgendeiner Stelle im Programm bei einem Variablennamen. Dann läuft es natürlich nicht korrekt und gerade solche Fehler sind schwer zu finden. Man kann diese strenge Überprüfung auf korrekt definierte Variablen- und Unterprogrammnamen mit der Anweisung

```
use strict;
```

im Programm erreichen. Andererseits hat man in Perl für die vielen einfachen, alltäglichen Probleme eine Programmiersprache, in der man einfach und sofort sagen kann, was man tun will. Es genügt beispielsweise die Zeile

```
print "Hello world!\n";
```

und schon wird `Hello world!` auf der Konsole ausgegeben. Der *Kontext*, in dem ein Literal oder eine Variable verwendet wird, wird durch die verwendeten Operatoren erzwungen. Es gibt zunächst keinen Unterschied zwischen Zahl und Zeichenkette. Paßt der Kontext, wird eine Zeichenkette als Zahl interpretiert.

Zum Beispiel ergibt „33“ + „44“ die Zahl 77. Dagegen ergibt „33“ . „44“ die Zeichenkette „3344“.

10.4.5 Variablen

Eine Variable ist ein Behälter für Werte, genauer ein Behälter mit einem Namen, so dass man ihn (und seinen Inhalt) wiederfinden kann. Dabei gibt es Variablen, die nur einen Wert enthalten können, genannt *Skalar*, und Variablen, die mehrere Werte enthalten können, genannt *Array*. In Perl werden die verschiedenen Arten von Variablen (= Datentypen) durch spezielle Zeichen markiert:

Datentypen des Perl-Interpreters

Typ	**Zeich.**	**Beispiel**	**bezeichnet:**
Skalar	$	$count	Einzelner Wert (Zahl oder String)
Array	@	@namen	Liste von Werten, über numerischen Index ansprechbar
Hash	%	%data	Assoziatives Array aus Wertepaaren, erster Wert ist Zugriffsschlüssel
Unter-programm	&	&convert	Aufrufbarer Perl-Code
Typeglob	*	*sonstwie	Alles was „sonstwie“ genannt wird

Skalare Variablen

Zuweisungen an Skalare erfolgen wie in den meisten Programmiersprachen mit dem „=“-Operator. Einer Skalar-Variablen kann dabei jeder skalare Wert zugewiesen werden (Integer, Fließpunktzahlen, Strings und sogar Referenzen auf andere Variablen oder Objekte). Es gibt verschiedene Möglichkeiten, die Werte für die Zuweisung zu erzeugen.

Will man Variablen einen ganzzahligen Wert sedezimal oder oktal zuweisen, muss man „0xzzz“ (sedezimal) bzw. „0zzz“ (oktal) schreiben. Wie bei C sind Konstante mit führender Null immer oktal.

Wie in der Linux-Shell kann man auch in Perl verschiedene Quoting-Mechanismen benutzen:

- Doppelte Anführungszeichen (double quotes) `" ..."` fügen Variablen-Inhalte in den String ein und lösen Backslash-Ausdrücke auf. Das folgende Beispiel gibt „Hallo Welt“ gefolgt von Zeilenwechsel aus:

```
$txt = "Hallo";
print "$txt Welt\n";
```

- Einfache Anführungszeichen (single quotes) `'...'` unterdrücken jegliche Interpretation von Variablen und Backslash. Das folgende Beispiel gibt „$txt Welt\n“ aus:

```
$txt = 'Hallo';
print "$txt Welt\n";
```

- Rückwärts gerichtete Anführungszeichen (backquotes) `...` führen ein externes Programm aus und liefern seine Ausgabe zurück, so dass alle Zeilen der Programmausgabe als einzelner String zur Verfügung stehen.

Die Anführungszeichen sind nicht die einzigen Quotes, alternativ kann man verwenden:

Quotes in Perl

Standard	Generic	Bedeutung
' '	q//	Zeichenkette
" "	qq//	Zeichenkette
` `	qx//	Kommando
( )	qw//	Wortliste
/ /	m//	Pattern matching

Beispiele für Zuweisungen:

```
$antwort = 42;                          \# Integer-Wertzuweisung
$x = 0xAFFE;                            \# Sedezimalzahl
$o = 0773;                              \# Oktalzahl
$pi= 3.14159265;                        \# Gleitpunktzahl-Wertzuweisung
$groesse = 7.678e12;                    \# Desgleichen in wiss. Notation
$tier =  "Kamel";                       \# String (das Perl-Maskottchen)
$satz =  "Ich liebe mein $tier";        \# String mit Ersetzung
$preis = "Die Variable heisst $tier";   \# String ohne Ersetzung
$gesamtpreis = $anzahl * $einzelpreis;  \# Ausdruck
$cwd = `ls /etc`;                       \# Kommandoausgabe
$exit = system("ls /etc");              \# numerischer Kommandostatus
```

Uninitialisierte Variablen werden bei Bedarf automatisch erzeugt. Nach dem Prinzip der minimalen Überraschung werden sie mit Nullwerten gefüllt, entweder 0 oder „0“. Je nach Verwendung werden Variablen automatisch als Strings, Zahlen oder boolsche Werte interpretiert. Dabei gelten Nullwerte als „false“, alle anderen Werte hingegen als „true“ (auch z. B. „00“).

Verschiedene Operatoren erwarten bestimmte Werte als Parameter. Man nennt dies „sie stellen einen skalaren Kontext zur Verfügung“. Dies kann auch genauer spezifiziert werden als numerischer Kontext, String-Kontext oder boolscher Kontext. Innerhalb sinnvoller Grenzen wandelt Perl die Daten automatisch in den erforderlichen Kontext um. Zum Beispiel bei:

```
$var = '123';
print "$var + 1 \n";
```

Der ursprüngliche Wert von $var ist ein String, er wird jedoch zu einer Zahl umgewandelt, um 1 dazuzuaddieren. Beim anschließenden „print“ wird der Inhalt

von $var für die Ausgabe wieder in einen String umgewandelt. Das Zeilenende „\n“ ist ebenfalls im String-Kontext, da es aber bereits ein String ist, entfällt die Umwandlung.

Vordefinierte Skalare

Die folgende Liste ist nicht vollständig, sondern enthält nur einige wichtige vordefinierte Skalare.

$_ Die bekannteste vordefinierte Variable in Perl. Sie enthält den jeweils aktuellen Wert (Eingabe, Suche usw.).

```
@Zahlen = (1..10);   # Zahlenarray, siehe unten
for(@Zahlen)
 { print $_, "\n"; }
```

$. Enthält die aktuelle Zeilennummer der zuletzt eingelesenen Datei

```
open(DATEI, "$$")
  { print $_; }
print $., " Zeilen gelesen";
close(DATEI);
```

$/ Enthält den eingestellten Eingabe-Separator. Kann geändert werden - auch mehrere Zeichen sind erlaubt.

```
$/ = "ENDE"; # statt \n ist nun }ENDE} das Zeilenende
$Eingabe = <STDIN>;
print $Eingabe;
```

$] Enthält die Versionsnummer des verwendeten Perl-Interpreters

$! Enthält die aktuelle Fehlermeldung oder Fehlernummer, sofern ein Fehler aufgetreten ist

```
 open(DATEI, "<nixda.txt") || print $!;
```

$0 Enthält den Dateinamen des Perl-Scripts, das gerade ausgeführt wird

$$ Enthält die Prozess-ID des Perl-Scripts, das gerade ausgeführt wird

```
$tempdatei = "Data." . $$;
```

$^T Enthält den genauen Zeitpunkt (Millisekunden), zu dem das Script gestartet wurde

```
...  # langes Programm
\$Start = $^T; $Jetzt = time;
print "Laufzeit: }, $Jetzt - $Start, } ms\n";
```

$\& Enthält nach Anwenden eines regulären Ausdrucks den Wert, auf den das Suchmuster passte

```
$Satz = "Katz und Maus";
$Satz =\textasciitilde s/\bund\b.*/;
print $&;
```

$+ Enthält nach Anwenden eines regulären Ausdrucks mit Klammern den Inhalt der Klammer, die mit dem letzten Suchmuster übereinstimmte

```
$Satz = "Katz und Maus";
$Satz =~ /(\bund\b).*/;
print $+;
```

$*n* ($1, $2, ...) Enthält nach Anwenden eines regulären Ausdrucks mit Klammern die Werte der Klammern 1, 2 usw., auf die das in der jeweiligen Klammer definierte Suchmuster passte

```
$Satz = "Hund und Katz und Maus";
$Satz =~ /(\bund\b).*/;
print "$1 $2";
```

10.4.6 Operationen und Zuweisungen

Perl kennt auch die üblichen arithmetischen Operatoren (Addition „+“, Subtraktion „-“, Division „/“, Modulo-Funktion „%“, Multiplikation „*“, Exponentation „**“ usw.). Für Zeichenketten gibt es noch den Append-Operator „.“ und den Repeat-Operator „x“. Die Zuweisungsoperatoren (einschließlich Autoinkrement und Autodekrement) entsprechen denen von C. Ebenfalls von C stammen die Vergleichsoperatoren. Bei Ausdrücken gilt „Punkt vor Strich“ und natürlich haben runde Klammern Vorrang.

```
$a = 1 + 2;   \# Addiere 1 und 2 und speichere Resultat in $a
$a = 3 - 4;   \# Subtrahiere 4 von 3 ...
$a = 5 * 6;   \# Multipiziere 5 und 6
$a = 7 / 8;   \# Dividiere 7 durch 8 ($a = 0.875)
$a = 9 ** 10; \# Neun hoch zehn
$a = 5 % 2;   \# 5 MOD 2
++$a;         \# Inkrement $a, Rückgabe von $a
$a++;         \# Rückgabe von $a, Inkrement $a
--$a;         \# Dekrement $a, Rückgabe von $a
$a--;         \# Rückgabevon $a, Dekrement $a
```

Für Strings gibt es unter anderem noch die beiden Operatoren:

```
$a = $b . $c; \# Konkateniere $b und $c
$a = $b x $c; \# Füge $b $c-mal zusammen
```

Verkürzte Zuweisungen:

```
$a += $b; \# $a = $a + $b
$a -= $b; \# $a = $a - $b
$a .= $b; \# $a = $a . $b
```

Vergleichsoperatoren:

Operation	**Zahlen**	**Strings**
gleich	==	eq
ungleich	!=	ne
kleiner	>	gt
kleiner/gleich	<=	le
größer/gleich	>=	ge
Vergleich	<=>	cmp

Beim Vergleich (letzte Zeile) liefert der Ausdruck -1, 0 oder +1, je nachdem, ob das linke Argument kleiner, gleich oder größer als das rechte ist.

Vergleich mit regulären Ausdrücken:

„wort“ in $X enthalten:	`$X =~/wort/`
„wort“ in $X nicht enthalten:	`$X !~/wort/`

Logische Operatoren:

<table>
<tr><td>&& bzw. <code>and</code>:</td><td>UND-Verknüpfung</td></tr>
<tr><td>|| bzw. <code>or</code>:</td><td>ODER-Verknüpfung</td></tr>
<tr><td>! bzw. <code>or</code>:</td><td>Negation</td></tr>
</table>

Bit-Operatoren:

<table>
<tr><td>>></td><td>verschiebt Bits nach rechts</td></tr>
<tr><td><<</td><td>verschiebt Bits nach links</td></tr>
<tr><td>¿</td><td>bitweise Und-Verknüpfung</td></tr>
<tr><td>|</td><td>bitweise Oder-Verknüpfung</td></tr>
<tr><td>ˆ</td><td>bitweise exklusive Oder-Verknüpfung</td></tr>
<tr><td>!</td><td>bitweise Negation</td></tr>
<tr><td>˜</td><td>Komplementbildung</td></tr>
</table>

Operatorenrangfolge

Prio.	Op.	Beschreibung
1	`,`	Aneinanderreihung
2	`= += -= *= /=`	Zuweisung
	`&= ^= |=`	
3	`? :`	Entweder-Oder-Bedingung
4	`..`	Bereichsdefinition in Listen
5	||	logisches Oder
6	`&&`	logisches Und
7	`| ^`	bitweises Oder – inklusiv/exklusiv
8	`&`	bitweises Und
9	`== !=`	Gleichheit/Ungleichheit
	`eq ne`	Gleichheit/Ungleichheit
10	`< <= > >=`	Vergleich größer/kleiner
	`lt le gt ge`	
11	`>> <<`	Verschieben von Bits
12	`+ - .`	Addition, Subtraktion, Zeichenkettenaddition
13	`* / % x`	Multiplikation, Division,
		Modulo-Division, Zeichenkettenwiederholung
14	`=~ !~`	Übereinstimmung bei regulären Ausdrücken
15	`**`	Potenzierung
16	`! ~ -`	Bitoperatoren
17	`++ --`	Inkrementieren, Dekrementieren

Die folgenden Programmzeilen sollen *Apfel and Birnen* ausgeben:

```
$a = 'Apfel';
$b = 'Birnen';
print $a . ' and ' . $b;
```

Es wäre schöner, nur einen String im *print statement* zu verwenden, aber mit

```
print '$a and $b';
```

erhalten wir *$a and $b*, was nicht unserem Wunsch entspricht. Die Verwendung doppelter Anführungszeichen veranlasst Perl, die Variablen aufzulösen:

```
print "$a and $b";
```

Daneben werden auch Spezialzeichen wie **\n** (Newline) und **\t** (Tabulator) aufgelöst. Im Folgenden sind alle diese Sonderzeichen aufgelistet, die auch bei den Ausgabefunktionen `print` und `printf` eingesetzt werden können:

\n	neue Zeile
\r	Wagenrücklauf
\f	Zeilenvorschub

\t	Tabulator
\a	Signalton
\e	Escape
\000-\377	Oktalcode (ASCII)

Arrays

Bestimmte Arten von Variablen können mehrere Werte aufnehmen, die logisch miteinander verknüpft sind: Arrays und Hashes. Sie verhalten sich in vieler Hinsicht wie Skalare, denn sie werden bei Bedarf leer erzeugt. Wenn ihnen ein Wert zugewiesen wird, so stellen sie auf der rechten Seite der Zuweisung einen „Listen-Kontext" zur Verfügung.

Ein Array ist eine geordnete Liste von Skalaren, auf die über die Position des Skalars in der Liste (Index) zugegriffen wird. Diese Liste kann Zahlen, Strings oder beides enthalten (sie kann sogar Referenzen auf andere Arrays enthalten, was den Aufbau multidimensionaler Arrays ermöglicht). Die Indexzählung beginnt immer bei 0. Um einem Array einen Listenwert zuzuweisen, gruppiert man einfach die Variablen mit Klammern zusammen:

```
@feld = ();                         \# leeres Feld
@Farben = (rot, gelb, gruen);       \# drei Zeichenketten
@krimskrams = (1, 2, 3, "hallo");   \# gemischt belegt
```

Man kann ein Feld auch in einem Listen-Kontext benutzt, wie z. B. auf der rechten Seite einer Listen-Zuweisung. Auf diese Weise kann man Skalarvariablen aus einem Array mit Werten belegen, z. B. :

```
($oben, $mitte, $unten) = @farben;
```

Wie in C, so werden auch in Perl Arrays nullbasiert indiziert. Der Zugriff auf einzelne Komponenten eines Arrays erfolgt durch einen in eckige Klammern eingeschlossenen Indexausdruck, zum Beispiel:

```
$farbe[0] = rot;
$farbe[1] = gelb;
$farbe[2] = gruen;
```

Der Indexausdruck kann jede beliebige Operation sein, die ein Integer als Ergebnis hat, also auch z. B. „$farbe[$i]" oder „$farbe[$i - $j]". Beachten Sie, dass bei Zugriff auf einzelne Komponenten der Klammeraffe wieder dem Dollarzeichen weicht. „@farbe" würde nämlich das ganze Feld referieren, so erhält man mit „print @farbe" als Ausgabe „rotgelbgruen". Der letzte Index eines Feldes wird durch die skalare Variable „$#feldname" adressiert. Im obigen Beispiel liefert $#farbe den Wert 2. Einen recht verblüffenden Effekt von Perl liefert die folgende Zuweisung:

```
$z = @farben;
```

Der Perl-Interpreter liefert den einzigen sinnvollen skalaren Kontext für ein Array, nämlich die Anzahl der Elemente. $z ist also gleich „$#farben + 1". Der skalare Kontext kann auch durch `$z = scalar(@farben)` erzwungen werden.

Des Weiteren kann in Perl der Index auch von hinten abgezählt werden. Im obigen Beispiel hätte die Anweisung:

```
print $farben[-2];
```

„gruen" ausgegeben. Das letzte Element könnte man also auch als `$farben[-1]` schreiben. Bei der Zuweisung können auch wieder Listen auf der rechten Seite auftauchen:

```
@colors = (schwarz, weiss, @farben, cyan, magenta);
```

Das Feld hat nun die Werte:

```
$colors[0] = schwarz
$colors[1] = weiss
$colors[2] = rot
$colors[3] = gelb
$colors[4] = gruen
$colors[5] = blau
$colors[6] = cyan
$colors[7] = magenta
```

Da Arrays geordnet sind, kann man daüber auch die Stack-Operationen „push" und „pop" ausführen. Perl betrachtet das Ende eines Arrays als Spitze eines Stacks, mit „push @array $skalar" wird ein Wert an ein Array angehängt. Man kann so ein Array sukzessive aufbauen. Mit dem internen Kommando „sort" kann eine Liste sortiert werden. Aber auch ohne „push" sind Arrays beliebig erweiterbar.

```
@feld = (null, eins, zwei);
$feld[$#feld+3] = fuenf;
```

Nun gilt:

```
feld[0] = null;
feld[1] = eins;
feld[2] = zwei;
feld[3] ist undefiniert;
feld[4] ist undefiniert;
feld[0] = fuenf;
```

Beispiel: Eine Liste sortiert ausgeben, Doubletten entfernen

```
@list = sort(@list);
$old = \qq{";
foreach $element (@list)
```

```
{
if ($element ne $old)
  { print $element, "\n"; }
$old = $element;
}
```

Array-Funktionen

Die folgende Tabelle zeigt die wichtigsten Array-Funktionen. Auf einige wird bei den Zeichenkettenfunktionen nochmals eingegangen. Mit „Liste“ wird in der Tabelle ganz allgemein eine Liste bezeichent. Die kann ein Array sein, aber auch eine geklammerte Liste von Skalaren, z. B. `($x, $y, $Z)`.

Array-Funktionen von Perl

Funktion	Beispiel und Bedeutung
grep	`@gefunden = grep(Muster, Liste);` Sucht alle Elemente der Liste, die den regulären Ausdruck *Muster* erfüllen
splice	`Liste = splice(@array,skip,anz,Neueliste);` Baut *anz* Elemente von *Neueliste* ab der Stelle *skip* in das Array ein und übergibt die so entstandene Liste ans Ergebnis
join	`$skalar = join(@array);` Fügt alle Elemente der Liste zu einem Skalar zusammen
split	`Liste = split(Muster,$skalar);` Teilt das Skalar an den durch das Muster identifizierten Stellen auf und bildet aus den einzelnen Elementen eine Liste
chop	`chop(Liste); $skalar = chop(Liste);` Holt das letzte Element aus einer Liste
push	`push(@array,Liste);` Fügt die Liste am Ende vom Array an
pop	`$skalar = pop(@array);` Holt ein Element aus der Liste
sort	`@sortiert = sort(Liste);` Sortiert eine Liste aufsteigend
reverse	`@sortiert = reverse(Liste);` Sortiert eine Liste absteigend
shift	`$skalar = shift(Liste);` Liefert das erste Element einer Liste
unshift	`$anz = unshift(@array,Liste);` Fügt die Liste am Anfang des Arrays ein. Das Ergebnis liefert die Anzahl der Elemente des neuen Arrays

Listen können auch verwendet werden, um gleichzeitig mehrere Zuordnungen von skalaren Variablen vorzunehmen:

```
($a, $b) = ($c, $d);            \# Aequivalent von $a = $c; $b = $d;

($a, $b) = @array;              \# $a und $b sind die ersten
                                \# beiden Elemente von @array.
```

```
($a, @somearray) = @array;       \# $a ist das erste Element von @array
                                 \# @somearray sind die übrigen

(@somearray, $a) = @array;       \# @somearray ist @array und
                                 \# $a ist nicht definiert.
```

Weil in Perl immer der Kontext eine Rolle spielt, ist es nicht erstaunlich, dass die folgenden Ausdrücke alle unterschiedliche Resultate erzeugen:

```
print @food;            \# Das ganze Array
print "@food";          \# mit doppelten Anführungszeichen
print "@food.''";       \# in skalarem Kontext (Konkatenation)
```

Übrigens funktioniert auch eine Zuweisung im skalaren Kontext, z. B. bei `$a = "@array"`.

Hashes

Ein „Hash" oder „assoziatives Array" ist eine Liste von Skalaren, auf die nicht über einen Index, sondern über einen eindeutigen Schlüssel-String zugegriffen wird. Ein Hash speichert pro Listen-Element zwei Werte: einen Namen und einen Wert. Werte innerhalb eines Hashes werden also über den zugehörigen Namen „assoziiert". Hashes sind intern über so genannte Hash-Tabellen implementiert (daher der Name) und deshalb sehr effizient. Beim Zuweisen von Werten an Elemente des Hashes muss ebenfalls der Schlüssel angegeben werden. Der Name eines Hashes wird durch das Prozentzeichen eingeleitet. Elemente innerhalb eines Hashes werden über Dollarzeichen angesprochen, wobei der Name in geschweifte Klammern gesetzt wird. Beispiel:

```
%Daten = ("Name","Meier",
          "Vorname","Hans",
          "Ort","Berlin"
         );
print $Daten{'Name'}, " wohnt in ", $Daten{'Ort'}";
```

Dies ist eine Form, den Hash mit Inhalten zu füllen. Eine weitere Möglichkeit ist (ab Perl 5):

```
%Daten = ("Name" => "Meier",
         "Vorname" => "Hans",
         "Ort" => "Berlin"
        );
```

Oder auch durch:

```
@Array = ("Name","Meier","Vorname","Hans",ört","Berlin");
%Daten = @Array;
```

Ein Hash kann in ein normales Array umgewandelt werden, indem er einfach dem Array zugewiesen wird. Umgekehrt kann ein Array in einen Hash umgewandelt werden. Idealerweise hat das Array eine gerade Anzahl von Elementen:

```
@info = %hash;          # @info ist ein normales Array.
$info[5];               # Das 5. Element von @info
%newhash = @info;       # %newhash ist ein assoziatives
                        # Array. Es ist der gleiche wie %hash
```

Für Hashes gelten im Übrigen die gleichen Regeln wie für gewöhnliche Listen. Mit der Funktion `keys` erhält man beispielsweise eine Liste der Schlüssel.

```
foreach $key (%hash)     # oder foreach $key (sort keys (%hash))
  { print $key, ": ", $hash{$key}, "\n"; }
```

Weitere Funktionen für Hashes finden Sie in der folgenden Tabelle.

Hash-Funktionen

Funktion	**Beispiel und Bedeutung**
keys	`@array = keys(%hash);` Liefert eine Liste der Indizes des Hash
values	`@array = values(%hash);` Liefert eine Liste der Inhalte des Hash
each	`@array = each(%hash);` Liefert alle Wertepaare in das Array
delete	`$skalar = delete($hash($key));` Löscht den durch $key identifizierten Hash-Eintrag
exists	`if (exists($hash($key))) ...` Liefert *wahr*, wenn ein Eintrag zum Key vorhanden ist

Vordefinierte Listen und Hashes

Die folgende Auflistung ist nicht vollständig, sondern enthält nur einige wichtige vordefinierte Listen- und Hash-Variablen.

@_ Parameter, die beim Aufruf eines Unterprogramms übergeben wurden. Innerhalb des Unterprogramms sind die übergebenen Parameter mit `$_[0]` (= erster Parameter), `$_[1]` (= zweiter Parameter) usw. ansprechbar.

```
&Sprich("We are the champion!");

sub Sprich
  { print $_[0]; }
```

@ARGV Parameter, die beim Aufruf des Perl-Scripts auf der Kommandozeile übergeben wurden

```
for($i = 0; $i <= $#ARGV; $i++)
 { print $ARGV[$i], "\n"; }
```

%ENV Enthält die Umgebungsvariablen, wie Sie dem Perl-Interpreter bekannt sind. Das nebenstehende Beispiel gibt die Elemente der in diesem Hash gespeicherten Daten aus.

```
print "User $ENV{'USER'} mit ";
print "Homedirectory $ENV{'HOME'} \n";
```

Eine besondere Rolle unter den oben aufgelisteten Hashes spielen die Umgebungsvariablen. Linux kennt das Konzept von Umgebungsvariablen, welche es erlauben, Informationen über das System an Programme weiterzugeben. Zum Beispiel wird in der Variablen USER der Name des eingeloggten Benutzers gespeichert und in der Variablen HOME dessen Arbeitsverzeichnis. Perl stellt diese Variablen in dem assoziativen Array %ENV zur Verfügung. Die Schlüsselwörter dieses Arrays sind die Namen der Umgebungsvariablen. Somit wird das folgende Programmstück die aktuellen Werte aller Umgebungsvariablen ausgeben:

```
for(%ENV)
 { print $_, "\n"; }
```

10.4.7 Kontrollstrukturen

Es gibt den üblichen Satz der bekannten Kontrollstrukturen höherer Sprachen sowie ein paar Perl-Spezialitäten. Ähnlich wie bei C werden die Anweisungen unterhalb einer Kontrollstruktur in geschweiften Klammern zusammengefaßt. Im Gegensatz zu C sind geschweifte Klammern für Zweige auch bei nur einer einzigen Anweisung nötig.

Bedingte Anweisungen

```
if (Bedingung) { ... }       # Anweisung ausführen, wenn die
                             # Bedingung wahr ergibt

if (Bedingung)  { ... }      # ausführen, wenn wahr (s. o.)
else { ... }                 # andernfalls diesen Zweig ausführen

if (Bedingung) { ... }       # gestaffelte Verzweigung
elsif (Bedingung)  { ... }
elsif (Bedingung)  { ... }
   ...
else { ... }
```

```
unless (Bedingung) { ... }   # Anweisung ausführen, wenn die
                             # Bedingung falsch ergibt

unless (Bedingung) { ... }   # desgleichen zweiseitig
else { ... }
```

`unless` bedeutet „wenn nicht“, ist also ein `if` mit negierter Bedingung.

Beispiel:

```
for($i = 1; $i <= 5; $i ++) # (for-Schleife siehe unten)
  {
  if    ($i < 3)  { print "($i) 1 oder 2\n" }
  elsif ($i == 4) { print "($i) gleich 4\n" }
  elsif ($i > 4)  { print "($i) kann nur 5 sein\n" }
  else            { print "($i) keine der Bedingungen erfuellt\n" }
  }
```

Wie Sie sehen, werden Bedingung durch Vergleiche oder reguläre Ausdrücke dargestellt. Nach der Auswertung der Bedingungen bedeuten die leere Zeichenkette (`""`, `"0"`) und die 0, dass die Bedingung nicht erfüllt wurde (falsch), und alle anderen Werte, dass die Bedingung erfüllt wurde (wahr). So entspricht beispielsweise (<DATEI>) „not EOF(DATEI)“ (Dateien siehe später).

`if (/zeichenkette/)` bedeutet „wenn die Zeichenkette im gelesenen Datensatz enthalten ist“.

Schleifenkonstrukte

```
while (Bedingung) { ... };     # nichtabweisende Wiederholung

until (Bedingung) { ... };     # desgleichen, aber Bedingung
                               # gegenüber while negiert

do { ... } while (Bedingung);  # nichtabweisende Wiederholung

do { ... } until (Bedingung);  # nichtabweisende Wiederholung,
                               # Bedingung negiert

for (Anfangsbedingung; Laufbedingung; Veränderung) { ... }

foreach $var (@list) { ... };
```

Die `for`-Schleife ist eigentlich nur eine Abkürzung für

```
Anfangsbedingung;
while (Laufbedingung)
  {
  ...
  Veränderung;
  }
```

Beispiel:

```
$i = 1;
print "Wurzeln von 1 bis 10...\n\n";
while ($i <= 10)
 {
 print "Die Wurzel von ", $i, " ist ", sqrt($i), "\n";
 $i++;
 }
$i = 1;
print "Quadratzahlen von 1 bis 10...\n\n";
until ($i > 10)
{
print "Das Quadrat von ", $i, " ist ", $i * $i, "\n";
$i++;
}
```

Schleifen mit `for/foreach` sind vor allem für Fälle vorgesehen, in denen Anfangswert und Endwert einer Schleife von vorneherein feststehen. `for (@Liste) ...` durchläuft die Schleife für jedes Element der Liste (jeweils aktuelles Listenelement als $_). Zum Beispiel:

```
for((1,2,3,4,5,6,7,8,9,10)) { print $_."\n" }
foreach((1,2,3,4,5,6,7,8,9,10)) { print $_."\n" }
foreach(1..10) { print $_."\n" }
foreach $nr (1..10) { print $nr."\n" }

for($i=0;  $i &lt;= 10;  $i++)
 {
 print "Das Quadrat von ", $i, " ist ", $i * $i, "\n";
 }

@Alphabet = (A..Z);
for(@Alphabet)
 {
 print $_, " ist der ", $i+1, ". Buchstabe im Alphabet\n";
 $i++;
 }
```

Sprungbefehle

- **exit**
  Sofortiges Beenden des Perl-Programms. Hinter `exit` kann ein Return-Code (Wert 0 für O.K., Wert > 0 als Fehlercode) stehen.
- **goto Ziel**
  Springe sofort zu dem mit `Ziel:` markierten Befehl.

- **redo**
  Der aktuelle Schleifendurchgang wird abgebrochen. Der nächste Schleifendurchgang wird gestartet, jedoch ohne die Schleifenbedingung neu zu bewerten (z. B. kein Hochzählen eines Schleifenzählers bei `for`).
  `redo` innerhalb des Schleifenblocks verhindert, dass der continue-Block ausgeführt wird.

- **next**
  Der aktuelle Schleifendurchgang wird abgebrochen. Der nächste Schleifendurchgang wird gestartet und die Schleifenbedingung dabei ebenfalls ausgewertet.

- **last**
  bricht eine Schleife sofort vollständig ab.

- **continue**
  leitet einen eigenen Anweisungsblock ein, genauso wie eine Schleife. Innerhalb dieses Anweisungsblocks (continue { ... }) können Sie beliebige Anweisungen notieren. Ein continue-Block unmittelbar hinter einem Schleifenblock wird bei jedem Schleifendurchlauf ausgeführt.

Verkürzte Strukturen

Kommt nach `if`, `unless`, `while` oder `do` nur eine Anweisung, sind bei Perl ja trotzdem geschweifte Klammern (also ein Block) nötig. Dafür kennt Perl jedoch eine Verkürzung durch Umstellen. Statt

```
if ($a > $b)
  { $max = $a; }
```

kann man schreiben:

```
$max = $a if ($a > $b);
```

Das gilt analog auch für die anderen Programmstrukturen. Mit der do-Anweisung lassen sich zudem noch Anweisungsblöcke wie ein einziger Ausdruck betrachten und so auch in diesem Kontext einsetzen. Das Ergebnis eines solchen do-Blocks ist der Wert der letzten Anweisung des Blocks.

```
do { $a = 2; $b = 5; $a * $b }
```

verhält sich wie ein Ausdruck, der den Wert 10 liefert. Man kann den obigen do-Block beispielsweise in einen Vergleich einsetzen:

```
if (do { $a = 2; $b = 5; $a * $b } == 1)
  { print "It works!\n"; }
```

Ein Ausdruck hinter do kann auch als Name einer Datei mit einem Perl-Script betrachtet und dieses im Kontext des aktuellen Programms ausgeführt werden, z. B.:

```
$script = "einfuege.pl";
do $script;
```

10.4.8 Reguläre Ausdrücke

Perl beherrscht als eingebautes Feature reguläre Ausdrücke. Dabei gibt es zwei Klassen: die Mustervergleichs-Operatoren „/foo/“ bzw. „m/foo/“ und die Ersetzungs-Operatoren „s/foo/bar/“. In Vergleichen kann Mustervergleich (pattern matching) mit dem Operator „=~“ angefordert werden. Reguläre Ausdrücke in Perl basieren auf einem NFA (nicht deterministischer finiter Automat), der folgendermaßen vorgeht: Er merkt sich die Stellen, an denen mehr als eine Möglichkeit zu kontrollieren ist. Stellt er beim Testen einer Variante fest, dass der Gesamtausdruck nicht mehr zutrifft, geht er zurück zum „Scheideweg“ und prüft die Alternative. Erst wenn alle abgehakt sind, entscheidet der NFA, ob der Ausdruck zutrifft oder nicht. Durch dieses „Backtracking“ genannte Vorgehen beherrscht Perl nummerierte Rückbezüge wie s/(Eins) (Zwei)/\2\1/g. Hier sorgen die Klammern dafür, dass Perl sich jedes „Eins“ und jedes „Zwei“ merkt. Im zweiten Teil vertauscht dann \2\1 die beiden miteinander.

Perl versucht normalerweise, den frühesten Treffer im String zu finden. Kommt aber ein Quantifizierer wie * ins Spiel, will der NFA so viel wie möglich finden, er wird gierig (greedy). Dabei ist die „Gierigkeit“ stärker als die Links-Bindung.

Will man beispielsweise in HTML-Code einen bestimmten Tag erwischen, heißt der erste Versuch vermutlich: /<.*>/. übersetzt: „Suche beliebig viele (auch gar kein) Zeichen, umschlossen von spitzen Klammern.“ Was würde Perl nun in der Zeile

```
<B>Wir</B> sind die <B>Champions</B>!
```

finden? Alles von der ersten spitzen Klammer bis zur letzten vor dem Ausrufezeichen. Da ein Quantifizierer dabei ist, gilt nicht mehr das Suchen nach einem Treffer so weit links wie möglich (also das erste <B>), sondern es siegt die Gier und es bleibt nur das Ausrufezeichen übrig. Perls Gierigkeit lässt sich jedoch durch ein hinter + oder * gesetztes Fragezeichen beschränken. Benutzt man im obigen Beispiel <.*?>, wird es <B> finden. In solchen Fällen hilft ebenfalls: /<[^>]+>/. Dieser Ausdruck sucht ein <, dann etwas, was kein > ist, davon mindestens eines, schließlich ein >. Ähnlich geht man zum Beispiel vor, um Worte in Anführungszeichen zu finden: /"[^"]+"/ erledigt diesen Job besser als /".*"/. Die Übersicht in den Tabellen 10.1 und 10.2 fasst die verschiedenen Möglichkeiten zusammen. Bei Perl werden reguläre Ausdrücke in Verbindung mit den folgenden Operatoren verwendet:

`=~/text/`: „text“ ist enthalten
`!~/text/`: „text“ ist nicht enthalten
`=~s/text1/text2/`: Textersetzung „text1“ durch „text2“
`=~tr/././`: Zeichenersetzung

Tabelle 10.1: Reguläre Ausdrücke in Perl, Teil 1

Ausdruck	Beschreibung
/string/	adressiert die nächste Zeile, die 'string' enthält (rückwärts suchen mit ?string?)
^	steht für den Zeilenbeginn `/^Meier/` adressiert Zeile, die mit „Meier“ beginnt
$	steht für das Zeilenende `/Meier$/` adressiert Zeile, die mit „Meier“ endet. `/^$/` adressiert die nächste Leerzeile
`[ ]`	definiert einen Buchstaben aus dem Bereich in `[ ]`. Beginnt der Bereich mit `^`, wird nach einen Zeichen gesucht, das **nicht** im Bereich enthalten ist (Bei Dateinamen war dies das !-Zeichen). `[ABC]`: einer der Buchstaben A, B oder C `[A-Z]`: Großbuchstaben `[A-Za-z]`: alle Buchstaben `[^0-9]`: keine Ziffer
`.`	der Punkt steht für ein beliebiges Zeichen
*	der Stern „*“ steht für eine beliebige Folge des **vorhergehenden** Zeichens (auch null Zeichen!). `a*`: Leer-String oder beliebige Folge von „a“ `aa*`: eine beliebige Folge von „a“ (mindestens eines) `[a-z]*`: Leer-String oder eine beliebige Folge von Kleinbuchstaben `[a-z][a-z]*`: eine beliebige Folge von Kleinbuchstaben (mindestens einer) `.*`: jede beliebige Zeichenfolge

Beispiele:

Tabelle 10.2: Reguläre Ausdrücke in Perl, Teil 2

Ausdruck	Beschreibung
*?	Der Stern „*“ ist recht „gefräßig“ (greedy), d. h. es wird versucht, maximal viele Zeichen in den reg. Ausdruck einzuschließen. Bei der Zeichenkette „aaa:bbb:ccc“ würde der Ausdruck „.*:“ die Zeichenkette „aaa:bbb:“ finden. Durch das nachgestellte Fragezeichen wird diese Eigenschaft umgekehrt, es wird die minimale Teilzeichenkette genommen - also im obigen Beispiel „aaa:“.
+	Das Pluszeichen „+“ steht für eine beliebige Folge des **vorhergehenden** Zeichens, jedoch mindestens eines. `a+`: ein a oder beliebige Folge von a's
?	Nullmal oder einmal das **vorhergehende** Zeichen
\	Hebt den Metazeichen-Charakter für das folgende Zeichen auf `a*` steht für Leer-String oder beliebige Folge von a's. `a\*` steht für die Zeichenfolge „a*“.
(...)	Reguläre Ausdrücke können mit Klammern gruppiert werden. `([A-Za-z]*)` gruppiert beispielsweise ein Wort aus beliebig vielen Buchstaben, wobei auch ein leeres Wort (0 Buchstaben) dazugehört. Soll das Wort mindestens einen Buchstaben enthalten, muss man `([A-Za-z][A-Za-z]*)` oder `([A-Za-z]+)` schreiben. Die Anwendung solcher Gruppen wird weiter unten gezeigt.
\i	Referenzieren des i-ten Klammerausdrucks

```
if ($value =~ /Meier/)        # "Meier" in der Zeichenketten enthalten?
$value =~ tr/+/ /;            # Ersetze "+" durch Leerzeichen
$value =~ tr/A-Z/a-z/;        # Ersetze alle Grossbuchstaben
                              # durch Kleinbuchstaben
$value =~ s/Meier/Huber/;     # Ersetze einmal "Meier" durch "Huber"
$value =~ s/Meier/Huber/g;    # Ersetze alle "Meier" durch "Huber"
```

Der folgende Ausdruck ersetzt alle Zeichenfolgen, die auf das Muster „%-Zeichen, gefolgt von zwei Sedezimalziffern“ passen, durch den Buchstaben, der den ASCII-Wert der Hexzahl hat:

```
$value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C",hex($1))/eg;
```

Wie Sie sehen, kann der Ersetzungsteil auch einen Perl-Ausdruck enthalten. Für die Zeichenklassen hat Perl noch einige Abkürzungen:

Zeichen und Klassen

Zeichenklassen bei Perl-Ausdrücken	
.	ein beliebiges Zeichen außer Newline; passt im Singleline-Mode auf das \n-Zeichen
x	ein x
Wort	ein „Wort“
[xyz]	Zeichenklasse: trifft x, y oder z
[a-z]	Zeichenklasse: passt auf jedes Zeichen zwischen a und z
(Wort)	eingefangenes „wort“, wird in $i gespeichert; mit \i im Ausdruck verwendbar
(?:Wort)	Gruppieren, aber nicht Speichern
x(?=y)	x wenn y folgt; Klammern speichern nicht
x(?!y)	x wenn kein y folgt. Klammern speichern nicht
\w	Wortzeichen
\w+	ganzes Wort (Wort mit folgendem Leer- oder Satzzeichen)
\W	Nicht-Wortzeichen: Satzzeichen, Leerzeichen und so weiter
\s	Leerraum: Leerzeichen, Tabulator, Newline
\S	alles, was kein Leerraum ist: Buchstaben, Ziffern etc.
\b	Wortgrenze
\d	Ziffer

Dazu gehören die folgenden Quantifizierer, die durch ein nachgestelltes Fragezeichen von „greedy“ auf „genügsam“ geschaltet werden können.

Quantifizierer bei regulären Ausdrücken	
*	vorangehender Ausdruck beliebig oft, auch gar nicht
+	vorangehender Ausdruck beliebig oft, mindestens einmal
?	vorangehender Ausdruck höchstens einmal
{n}	vorangehender Ausdruck genau n-mal
{n,}	vorangehender Ausdruck n-mal oder häufiger
{n,m}	vorangehender Ausdruck n- bis m-mal

Schließlich gibt es noch reguläre Ausdrücke mit Alternativen, dabei dient das Pipe-Symbol als Trenner:

- /a |b/ findet „a“ oder „b“, entspricht also [ab].
- /aus|ohne/ findet „aus“ und „Hausbau“, aber auch „ohne“ und „Bohne“.
- /a|bc|d/ findet „a“ oder „bc“ oder „d“.
- /(a|b)(c|d)/ findet „ac“ oder „ad“ oder „bc“ oder „bd“.

Sonderzeichen

<table>
<tr><th>Zeichen</th><th>Beschreibung</th></tr>
<tr><td>|</td><td>oder, auf Zeichen und Gruppen anwendbar: (ganz)—(gar nicht)</td></tr>
<tr><td>^</td><td>Anfang der Zeile oder des Ausdrucks</td></tr>
<tr><td>$</td><td>Ende der Zeile oder des Ausdrucks</td></tr>
<tr><td>[^x]</td><td>alles, was nicht „x“ ist</td></tr>
<tr><td>\</td><td>Backslash, nimmt Sonderzeichen die Spezialbedeutung</td></tr>
<tr><td>\U</td><td>alle Zeichen bis \E in Großbuchstaben</td></tr>
<tr><td>\L</td><td>alle Zeichen bis \E in Kleinbuchstaben</td></tr>
<tr><td>\Q</td><td>alle Sonderzeichen bis \E mit \ schützen</td></tr>
<tr><td>\E</td><td>Ende von \U, \L, \Q</td></tr>
</table>

Achtung: Die Metazeichen `+ - ? . * ^ $ ( ) [ ] { } | \` müssen durch ein davorstehenden Backslash (\) geschützt werden, wenn sie im Suchstring auftauchen.

Beispiele:

```
# Alles in Grossbuchstaben
$line =~ tr/a-z/A-Z/;

# Alles in Kleinbuchstaben
$line =~ tr/A-Z/a-z/;

# fuehrende Leerzeichen entfernen
$line =~ s/^\s+//;

# Leerzeichen am Ende entfernen, auch \n
$line =~ s/\s+$//

# Testen, ob eine Jahreszahl vierstellig ist
# Format: nn/nn/nnnn oder nn-nn-nnnn
# true, falls OK
($line =~ m/[0-9]{2}[\/|-][0-9]{2}[\/|-][0-9]{4}/)

# Entfernt HTML-Tags aus dem Text
$line =~ s/(<([^>]+)>)//ig;
```

Perls reguläre Ausdrücke können vorausschauen, ob ein String passen könnte (lookahead). Mit */Haus(?=bau)/* findet der Interpreter „Haus“ nur dann, wenn „bau“ folgt. Schließt sich an „Haus“ jedoch „herr“ an, trifft der Ausdruck nicht

mehr zu. Das Ganze darf man auch verneinen: „Finde jedes 'Haus', aber nur, wenn dem kein 'bau' folgt“: `/Haus(?!bau)/`.

Für nummerierte Rückbezüge kennt Perl zwei Schreibweisen: Die Variablen `$1`, `$2`, `$3` und so weiter enthalten jeweils den Wert des in der ersten, zweiten, dritten ... einfangenden Klammer gefundenen Musters. Sie lassen sich irgendwo im Programm verwenden. `\1`, `\2` sind dagegen Bestandteile der RegEx-Maschine. In ihnen steht ebenfalls der in den einfangenden Klammern gefundene Wert. Die Anzahl der Rückbezüge ist in beiden Fällen unbegrenzt. Vorsicht ist jedoch geboten: Aufgrund der Art und Weise, wie die RegEx-Maschine den Ausdruck interpoliert und kompiliert, gibt es durchaus einen Unterschied zwischen `$1` und `\1`. Empfohlen wird, innerhalb des Ausdrucks nur den Rückbezug mittels `\1` zu verwenden, da Variablen wie `$1` erst später ausgewertet werden und beim ersten Interpretieren eines Ausdrucks möglicherweise noch gar keinen Wert besitzen. Neben `$1`, ... belegt Perl bei jeder RegEx-Auswertung einige Spezialvariablen neu: In `$&` findet sich immer der letzte gültige Treffer, in `$&´` alles, was vor ihm und in `$&‘` das, was nach ihm lag.

Perl erlaubt in seinen regulären Ausdrücken Ausdrücke und Funktionen, die einen korrekten String ergeben, wenn man den Modifizierer `e` benutzt. Er sorgt dafür, dass die RegEx-Maschine alle Variablen interpoliert, den Ausdruck übersetzt, `\1` usw. belegt und den Ersetzungsteil evaluiert. Beispiel:

```
...
$system =~ s/Betriebssystem/&os/eg;
...

sub os
 {
 $string = ‘uname -a‘;
 return $1 if ($string =~ /(^\w+)/)
 }
```

Hier wird „Betriebssystem“ durch den Rückgabewert des Unterprogramms *os()* ersetzt.

Operatoren, Modifizierer und Spezialvariablen

Operator	Beschreibung
m//	suche
s///	suche und ersetze
s+++	suche und ersetze, aber verwende + als Trenner
//g	jedes Vorkommen finden
//i	Groß- und Kleinschreibung ignorieren
//m	Multiline-Mode, ^ und $ passen auf logische Zeilenanfänge und -enden, der Punkt matched kein Newline
//s	Singleline-Mode, ^ und $ erkennen Anfang/Ende des gesamten Strings, . matched Newline
//sm	kombiniert: Logische Zeilen plus Newline
//x	Kommentare und Leerzeichen im Suchen-Teil erlaubt
s///e	Ausdruck im Ersetzungsteil erst evaluieren, dann ersetzen
$n	Wert der n-ten einfangenden Klammer; nur im Ersetzungsteil verwendbar
\1	Rückbezug auf die erste einfangende Klammer
$\&	letzter gefundener Treffer
$'	Text nach dem Treffer
$`	Text vor dem Treffer
$+	Treffer der letzten einfangenden Klammer
$/	Zeilentrenner (Input Record Separator)

Abschließend zwei Beispiele für häufig verwendete reguläre Ausdrücke. Der erste Ausdruck beschreibt eine E-Mail-Adresse (zumindest oberflächlich), um testen zu können, ob sich ein Sendeversuch überhaupt lohnt:

```
([\w\-\.]+)@([\w\-\.]+)
```

Die beiden Zeichenklassen (`[\w\-\+\._]`) vor und hinter dem „@" beschreiben einen Namen aus Buchstaben, Ziffern, dem Minuszeichen, dem Underline und dem Punkt. Das deckt sich ungefähr mit Usernamen bzw. Domainangaben. Eine Gültigkeit der Adresse ist jedoch nicht feststellbar.

Der zweite Ausdruck zeigt die Zerlegung einer URL in ihre Teilkomponenten:

```
($host,$port,$file) = ($url =~ m|http://([^/:]+):{0,1}(\d*)(.*)$|);
```

`http://` steht für sich selbst, dann kommt eine Zeichenfolge, die entweder mit „/" oder „:" endet, also die Hostangabe. „:0,1" deckt sich mit einem oder keinem Doppelpunkt. Die folgende Ziffernfolge (die auch leer sein kann) wird in $port gespeichert und der Rest der URL in $file. Das Ganze geht schief, wenn der Dateiname nur aus Ziffern besteht.

10.4.9 Unterprogramme

Unterprogramme ermöglichen die Strukturierung des Programms. Mehrfach im Programm benötigter Code muss nur einmal aufgeschrieben werden. Außerdem

erhöt die Verwendung von Unterprogrammen die Lesbarkeit und sie macht auch die Programmpflege einfacher. In Perl erfolgt die Definition mit dem „sub“-Schlüsselwort:

```
sub name
  { ... }
```

Zum Aufruf von Funktionen wird „&name“ oder „name()“ verwendet. Unterprogramme können auch einen beliebigen Wert zurückliefern, man nennt sie dann auch „Funktionen“. Damit ergibt sich folgender allgemeiner Aufbau:

```
sub myfunc
  {
  ... Anweisungsteil ...

  return returnvalue;
  }
```

Einem Unterprogramm können auch Parameter mitgegeben werden, es erfolgt jedoch im Gegensatz zu C keine formale Definition der Parameter. Ab Perl 5.003 darf auch ein Prototyp für die Parameter verwendet werden, indem die Zeichen „$“, „“ und „%“ als Platzhalter für Parameter vom jeweiligen Typ stehen, z. B. `sub myfunc($$)`. Hier ist dann der Aufruf nur mit zwei Skalarparametern erlaubt.

Eigenschaften von Funktionen:

- In Perl können Sie Ihre Funktionen an beliebigen Stellen im Programm definieren.
- Perl kennt keine formale Parameterliste in der Kopfdeklaration wie Pascal oder C/C++, sondern arbeitet mit einer variablen Anzahl schwach typisierter Parameter.
- In Perl können Sie mehrere Rückgabewerte zurückliefern. Des Weiteren kann der Kontext (wantarray: Liste oder Skalar) abgefragt werden.
- In Perl stehen alle Argumente, die an eine Funktion übergeben werden, als Parameterliste im Spezial-Array @_ zur Verfügung. Über @_ bzw. $_[0], $_[1], ... usw. sind sie erreichbar.
- Mit Perl ist ein einfaches Prototyping möglich (siehe oben).
- Perl besitzt etliche eingebaute Funktionen, die der Perl-Interpreter kennt und die vor ihrer Verwendung nicht deklariert werden müssen.

Vordefinierte Perl-Funktionen reichen von arithmetischen Funktionen und Funktionen für die Zeichenkettenmanipulation bis zu Datenbank-, Netzwerk- und Prozesskommunikations-Funktionen. Für den Aufruf einer Funktion gibt es in Perl drei Möglichkeiten:

```
myfunc (Parameterliste);
myfunc Parameterliste;
\&myfunc;
```

Im letzten Fall wird das Standardargument @_, also eine Liste aller Parameter übergeben. Dazu einige Beispiele:

Einfachster Fall ohne Wertübergabe:

```
sub zaehle
 {
 my $i;                    # lokale Variable $i
 $summe = 0;
 for ($i=1; $i<= 10; $i++ )
   {
   $summe = $summe + $i;
   }
 }

&zaehle;
print "Die Summe betraegt: $summe\n";
```

Wertübergabe ans Unterprogramm:

```
sub gibaus
 {
 foreach $listeninhalt (@_)
   { print $listeninhalt, "\n"; }
 }

@tabelle = (Tick, Trick, Track);
&gibaus(@tabelle);
```

Aufruf als Funktion, es werden Werte zurückgegeben.

```
sub gibaus
 {
 my(@liste) = @_; # Parameteruebergabe an lokales Array @liste
 foreach $listeninhalt (@liste)
   { print $listeninhalt, "\n"; }
 return $#liste;
 }

@tabelle = (Tick, Trick, Track);
$anzahl = &gibaus(@tabelle);
print "Tabelle hat: ", $anzahl+1, " Elemente\n";
```

Das Resultat einer Funktion ist immer das Resultat des letzten evaluierten Ausdruckes. Dieses Unterprogramm berechnet das Maximum von zwei gegebenen Eingabeparametern:

```
sub maximum
 {
 if ($_[0] > $_[1]) { $_[0]; }
 else               { $_[1]; }
 }
```

Der Aufruf sieht folgendermaßen aus:

```
$max = &maximum(37, 24);
```

Das Resultat einer erfolgreichen **print**-Anweisung ist immer 1. Man kann den Rückgabewert aber auch explizit mit `return` *Wert* an das aufrufende Programm zurückgeben.

Noch eine Funktion mit Parameterübergabe und Rückgabewert:

```
sub Rechne
  {
  # eval() betrachtet einen String als Rechenausdruck
  # und gibt das errechnete Ergebnis zurück.
  return eval($_[0]);
  }

$x = &amp;Rechne(1 + 2 * 3 + 4 * 5);
print $x, "\n";
```

Die einzelnen Parameter werden mit $_[0], $_[1], ... angesprochen und müssen deshalb nicht als formale Parameter definiert werden. Diese Parameter haben nichts mit der Spezialvariablen **$_** zu tun und man kann beides verwenden, ohne Namenskollisionen befürchten zu müssen.

Variablen, die innerhalb eines Unterprogramms implizit definiert werden, behalten nach Beenden des Unterprogramms ihre Gültigkeit und sind damit globale Variablen. Die Verwendung des gleichen Namens im Haupt- und Unterprogramm kann zu Fehlern führen. Die Lebensdauer einer Variablen kann aber auf den Block des Unterprogramms beschränkt werden, indem sie explizit mit „my“ oder „local“ als lokal definiert wird. Es wird empfohlen die ältere Methode mit „local“ nicht mehr zu verwenden.

Beispiel 1 (globale Variable):

```
my $x = 10;
&Rechne;
print "HP: ", $x, "\n";

sub Rechne
 {
 $x = 5;
 print "UP: ", $x, "\n";
 }
```

Beispiel 2 (lokale Variable):

```
my $x = 10;
&Rechne;
print "HP: ", $x, "\n";

sub Rechne
 {
 my $x = 5;
 print "UP: ", $x, "\n";
 }
```

Die folgende Subroutine testet, ob ein String in einem anderen enthalten ist, ohne die Leerzeichen zu berücksichtigen:

```
sub inside
 {
 my ($a, $b);                     # Erzeuge lokale Variable
 ($a, $b) = ($_[0], $_[1]);       # Param. zuordnen
 $a =~ s/ //g;                    # Leerzeichen aus lokalen
 $b =~ s/ //g;                    # Variablen loeschen
 ($a =~ /$b/ || $b =~ /$a/);      # Ist $b in $a  oder $a in $b?
 }

&inside("lemon", "dole money");
# true, weil        ^^ ^^^
```

Falls mehr als eine Variable mit `my` deklariert werden soll, müssen sie wie oben in Klammern angegeben werden (`my` ist eine Funktion!).

`local` war in Perl 4 die einzige Möglichkeit, lokale Variabeln zu deklarieren. Aus Kompatibilitätsgründen und für Spezialfälle ist `local` noch verfügbar. `my` ist die neuere (ab Perl 5) und effizientere Art, lokale Variablen zu deklarieren. Der Unterschied zwischen `my` und `local` ist subtil:

- `local` erzeugt einen temporären Wert für eine globale Variable (dynamic scoping),
- `my` deklariert dagegen eine echte lokale Variable (lexical scoping), welche nur im umgebenden Block bekannt ist.

Die Syntax ist bei beiden gleich. Eine Einschränkung gibt es nur bei den Variablennamen: Bei `my` sind nur alphanumerische Zeichen erlaubt! Um Spezialvariablen wie z. B. „$\“ zu lokalisieren, muss nach wie vor `local` verwendet werden.

Beispiel: Wochentagsberechnung

```
sub dayofweek
  {
```

```
# Liefert den Wochentag für ein Datum
# Eingabeparameter: Tag, Monat and Jahr.
#            z.B.: &dayofweek(18, 9, 2004);
#  Rueckgabewert: 0=Sonntag, 1=Montag, ...
my ($day, $month, $year) = @_;
my ($a, $y, $m, $dow);
my @daysinmonth = (0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31);
# Schaltjahr?
if ((($year%4 eq 0) && ($year%100 ne 0)) || ($year%400 eq 0))
  { $daysinmonth[2]++ ; }

if (($month < 1 || $month > 12) ||
    ($day < 1 || $day > $daysinmonth[$month]))
  {
  return(-1);
  }
else
  {
  $a = int((14 - $month)/12);
  $y = $year - $a;
  $m = $month + (12 * $a) - 2;
  $dow = ($day + $y + int($y/4) - int($y/100)
           + int($y/400) + int((31 * $m)/12)) % 7;
  return($dow);
  }
}
```

Vordefinierte Perl-Funktionen für Zeichenketten

- **sort**: Sortieren einer Liste. Funktionsaufruf: `sort(@Liste);` oder `sort @liste;`
- **split Pattern, Ausdruck**: zerlegt einen Zeichenketten-Ausdruck an einem bestimmten Zeichen und weist das Ergebnis einem Array zu. Der Funktionsaufruf `@liste = split(/ /, $string);` zerlegt eine Zeile in Worte.
- **join Ausdruck, Liste**: vereint die Elemente einer Liste zu einem String, wobei die Elemente durch das Ergebnis des Ausdrucks verbunden werden, z. B. `$s = join('-',@liste);`.
- **grep Ausdruck Liste**: Sucht alle Elemente einer Liste, die auf einen regulären Ausdruck passen, z. B. liefert `@result = grep(/M̂/, @liste);` alle Elemente von `@liste`, die mit „M“ beginnen. Beispiel:

```
print "Geben Sie eine Folge von Begriffen ein, getrennt durch Komma:\n";
$Eingabe = <STDIN>;
chomp $Eingabe;
@Liste   = split(/,/, $Eingabe);
@SortierteListe = sort(@Liste);
```

```
for(@SortierteListe)
 { print $_, "\n"; }
```

- **Teilstring = substr (String, Pos)**: liefert den Rest des Strings ab der Position „Pos“
- **Teilstring = substr (String, Pos, Anz)**: liefert „Anz“ Zeichen aus String ab der Position „Pos“
- **Pos = index(String1, String2)**: liefert die Position, an der „String2“ erstmals im „String1“ vorkommt. „String2“ kann auch nur ein Zeichen sein. Ist „Pos“ = 0, wurde String2 nicht gefunden.
- **Pos = rindex(String1, String2)**: liefert analog das letzte Vorkommen von „String2“ in „String1“
- **length (String)**: liefert die Länge des Strings.

Beispiel:

```
$Url = "http://www.netzmafia.de/scripten/sicherheit/index.html";
$Stop = index($Url,":");
$Protokoll = substr($Url,0,$Stop);
$Start = index($Url,"//") + 2;
$Domain = substr($Url,$Start);
$Stop = index($Domain,"/");
$Domain = substr($Domain,0,$Stop);
$Start = rindex($Url,"/") + 1;
$Dateiname = substr($Url,$Start);
$Url_Laenge = length($Url);
print "Protokoll.....: ", $Protokoll, "\n";
print "Domain-Adresse: ", $Domain, "\n";
print "Dateipfad.....: ", $Dateiname, "\n";
print "Laenge........: ", $Url_Laenge, "\n";
```

Perl besitzt Funktionen für die wichtigsten höheren mathematischen Berechnungen, darunter:

abs: Absolutwert
sqrt: Quadratwurzel
int: ganzzahliger Anteil
exp: Potenz zur Basis e
log: nat. Logarithmus
sin: Sinus
cos: Cosinus

Ebenso gibt es Befehle zum Ausführen von Kommandos auf Systemebene:

- `system("Kommando")`: Angegebenes Kommando ausführen, Rückgabewert ist der Returncode des externen Programms

- `exec("Kommado")`: Aktuelles Perl-Programm mit dem angegebenen Kommando überlagern, keine Rückkehr zum Perl-Programm
- `$var = ‘Kommando‘`: Angegebenes Kommando ausführen, in $var gelangt die Ausgabe des externen Programms

10.4.10 Dateien

Im Folgenden wird davon ausgegangen, dass das Perl-Script die passende Zugriffsrechte für Dateien und Verzeichnisse besitzt. Bevor man von einer Datei lesen oder auf eine Datei schreiben kann, muss man diese öffnen. Dabei wird ihr ein File-Handle zugeordnet, das für die weiteren Zugriffe verwendet wird. Beim Öffnen wird bereits angegeben, ob die Datei gelesen oder beschrieben werden soll. Zum Lesen muss die Datei bereits existieren. Soll eine Datei zum Schreiben geöffnet werden, die bereits existiert, so wird der alte Datei-Inhalt überschrieben. Existiert die zu beschreibende Datei noch nicht, wird eine neue Datei mit dem angegebenen Namen angelegt.

Zum Öffnen einer Datei dient der Befehl

```
open(FileHandle,’Dateibezeichnung und Öffnungsmodus’);
```

Als File-Handles verwendet man üblicherweise Namen, die nur aus Großbuchstaben gebildet werden, um sie von anderen Variablen zu unterscheiden. Es steht auch kein Dollarzeichen vor dem Handle. „STDIN“ und „STDOUT“ bezeichen die Filehandles für die Standardeingabe und -ausgabe.

```
open(SESAM, "filename");              # lesen aus existierender Datei
open(SESAM, "<filename");             # dito (explizit)
open(SESAM, ">filename");             # Datei erzeugen und schreiben
open(SESAM, ">>filename");            # an Datei anhängen
open(SESAM, "|output-pipe-command");  # Ausgabepipe erzeugen (s. unten)
open(SESAM, "input-pipe-command|");   # Eingabepipe erzeugen (s. unten)
```

Zum Schließen einer Datei verwendet man das Kommando

```
close(FileHandle);
```

Lesender Zugriff

Zum Lesen aus dem Filehandle wird der „<>“-Operator benutzt, damit liefert „<FILEHANDLE>“ immer die nächste Zeile aus der Datei. Am Datei-Ende liefert dieser Operator einen Nullwert als Ergebnis. Das Einlesen des Datei-Inhaltes kann zeilenweise erfolgen. Mit:

```
$Zeile = <SESAM>;
```

wird jeweils die nächste noch nicht gelesene Zeile eingelesen und der links stehenden Variablen als Wert zugewiesen. Man kann auch alle Zeilen auf einen Schlag lesen und sie als Werte einer Liste abspeichern:

```
@Zeilen = <SESAM>;
```

Man kann sogar noch weiter gehen und die gesamte Datei in einer einzigen Zeile abspeichern:

```
@Zeilen = <SESAM>;
$line = join(" ", @Zeilen);
```

Das ist dann ganz praktisch, wenn man in der gesamten Datei Ersetzungen oder Umformungen vornehmen will. Mit `print $line;` wird dann der gesamte Datei-Inhalt weggeschrieben. Die Newline-Zeichen bleiben dabei erhalten!

Das Öffnen einer Datei ist immer eine fehlerträchtige Situation, man sollte deshalb im Fehlerfall eine Fehlermeldung ausgeben, z. B.:

```
open(SESAM,'daten.dat') || die "Eingabedatei konnte nicht geoeffnet werden";
     ...
```

Der Befehl „die“ („stirb“) beendet das Programm mit einer Fehlerausgabe, er wird aufgerufen, falls das Öffnen der Datei fehlschlägt.

Perl liefert beim Einlesen eines Satzes auch das Zeichen, mit dem das jeweilige Ende des Datensatzes angegeben wird, üblicherweise ist dies Newline(\n). Wenn man dieses Zeichen nicht im Eingabe-String haben möchte, muss man es explizit entfernen. Der Befehl dazu ist:

```
 chop($Satz);
```

Er entfernt das jeweils letzte Zeichen der Eingabe. Manchmal ist der Gebrauch von chop ungünstig, da immer das letzte Zeichen des Strings entfernt wird, auch wenn es kein Newline ist. Da hilft

```
chomp($Satz);
```

Mit diesem Befehl wird ein Newline am String-Ende entfernt. Alle anderen Zeichen werden verschont.

Schreibender Zugriff

Zum Schreiben öffnet man eine Datei mit:

```
open(File-Handle,'>$Dateiname');
```

wenn ein möglicherweise vorhandener Datei-Inhalt überschrieben werden soll. Sollen Daten an eine vorhandene Datei angehängt werden, öffnet man die Datei mit:

```
open(File-Handle,'>>$Dateiname');
```

Die Ausgabe in die Datei erfolgt mit der print-Anweisung:

```
print SESAM  Ausgabedaten;
```

Geschrieben wird damit in die aktuelle Ausgabezeile, Zeilenwechsel müssen in Perl explizit angegeben werden. Für Linux-Dateien ist dies das Zeichen 'newline', mit der Ersatzdarstellung „\n“, bei DOS/Windows-Dateien gibt man „\r\n“ aus. Beachten Sie auch, dass im Gegensatz zu anderen Programmiersprachen kein Komma zwischen Filehandle und Ausgabedaten steht, sondern ein Leerraum.

```
open(LESEN,"alt.dat");
open(SCHREIBEN,"> neu.dat");

while(defined($i = <LESEN>))
  { print SCHREIBEN $i }

close(LESEN);
close(SCHREIBEN);
```

Ein- und Ausgabe mit Pipes

Man kann als Eingabestrom für ein Script auch die Ausgabe eines anderen Prozesses verwenden, die Standardausgabe des anderen Prozesses wird dann in einen Eingabestrom des Perl-Scripts gepiped.

```
open(SESAM, "ls -l |");
while ($Zeile = <SESAM>)
 {
 print $Zeile;
 }
```

Die Ausgabe des Linux-Kommandos 'ls -l' wird über den File-Handle SESAM zeilenweise eingelesen und auf die Standardausgabe ausgegeben. Das Kommando `ls` wirkt hier auf das Verzeichnis, in dem das Perl-Script ausgeführt wird.

Ebenso kann man auch einen Ausgabestrom in einen anderen Prozess pipen. Im folgenden Beispiel wird ein Ausgabestrom an ein Drucke-Kommando weitergereicht:

```
open(SESAM, "| lpr");
while (<>)
 {
 print SESAM $_;
 }
```

Beispiel: Mail versenden:

```
open(MAIL, "|/bin/mail plate\@netzmafia.de");
print MAIL "Dies ist ein Test\n";
close(MAIL);
```

Man beachte, dass hier der Klammeraffe maskiert werden muss, da er für sich selbst steht und keine Listennamen kennzeichnet.

Beispiel: Eine Datei kopieren:

```
open(INFILE, "eingabe");
open(OUTFILE,äusgabe");
while (<INFILE>)
 { print OUTFILE $_; }
close(OUTFILE);
close(INFILE);
```

Die Filehandles `STDIN`, `STDOUT` und `STDERR` sind vordefiniert. Per Default schreibt `print` auf `STDOUT` und `<>` liest von `STDIN`, (`<>` ist äquivalent zu `<STDIN>`).

Sperren des Dateizugriffes für andere Benutzer

Bei CGI-Scripts kann es sein, dass mehrere Webserver-Prozesse gleichzeitig laufen und daher auch mehrere Clients gleichzeitig auf eine Datei zugreifen. Solange beide nur lesen wollen, ist das unproblematisch, es können aber Konflikte entstehen, wenn mehrere Clients schreiben wollen. Mit dem „flock"-Befehl lassen sich die Zugriffe anderer Benutzer auf eine Datei sperren. Die Prozesse dieser Benutzer müssen solange auf den Zugriff warten, bis Ihr Prozess den Zugriff auf die Datei wieder zulässt. Der Befehl „flock" wird mit dem File-Handle der Datei und einer Zahl zur Spezifizierung der Sperre aufgerufen.

Num.	**Konstante**	**Beschreibung**
1	LOCK_SH	shared read lock. Offener Zugriffsschutz, wird meist beim lesenden Zugriff verwendet. Andere dürfen während der Sperre die Datei lesen, schreiben darf jedoch nur der Benutzer, der die Sperre veranlasst hat.
2	LOCK_EX	exclusive write lock. Exklusiver Zugriffsschutz, erlaubt anderen während der Sperre keinerlei Zugriff auf die Datei, auch keinen Lesezugriff
4	LOCK_NB	non blocking lock. Setzt lediglich einen Vermerk auf die Datei, ohne andere am Zugriff zu hindern (Nicht blockierende Sperre).
8	LOCK_UN	unlock. Hebt eine Sperre wieder auf

Dateifunktionen

Zum Arbeiten mit Dateien gibt es folgende Funktionen:

- Änderung des Namens: `rename()`
  Beispiel: `rename("alter_name","neuer_name");`
- Löschen einer Datei: `unlink()`
  Beispiel: `unlink("dateiname");`
- Erzeugen eines Verzeichnisses: `mkdir()`
  Beispiel: `$permissions = 0777; mkdir("dirname",$permissions);`
- Löschen eines (leeren) Verzeichnisses: `rmdir()`
  Beispiel: `rmdir("dirname");`
- Hard Link erzeugen: `link()`
  Beispiel: `link("filename", "linkname");`
- Soft Link erzeugen: `symlink()`
  Beispiel: `symlink("filename", "linkname");`
- Ändern der Zugriffsrechte: `chmod()`
  Beispiel:
  ```
  $permissions = 0755;
  chmod($permissions, "kap1.tex", "kap2.tex");
  ```
- Ändern des Besitzers: `chown()`
  Beispiel: `$uid = 100; $gid = 10; chown($uid, $gid, "datei");`
- Ändern des Zeitstempels: `utime()`
  Beispiel:
  ```
  $access = 812_000_000;  # letzter Zugriff
  $modifi = 822_000_000;  # letzte Aenderung
  utime($access, $modifi,"datei");
  ```
  Die Zeiten sind hierbei in Sekunden seit dem 1.1.1970 anzugeben. Die aktuelle Systemzeit kann über den Operator `time` abgefragt werden.

Datei-Attribute

-r	lesbar
-w	schreibbar
-x	ausführbar
-o	gehört dem Benutzer
-e	existiert
-z	leer
-s	nicht leer (liefert die Dateigröße in Bytes)
-f	ist eine normale Datei
-d	ist ein Directory
-l	ist ein symbolisches Link
-S	ist ein Socket
-p	ist ein FIFO
-b	ist ein Blockdevice
-c	ist ein Zeichengerät
-t	ist ein Ausgabegerät
-T	enthält Text
-B	enthält Binärdaten
-M	Zeit seit der letzten Änderung in Tagen
-A	Zeit seit dem letzten Zugriff

Besonderheiten der Konsole

Bei Ein- und Ausgabe über Bildschirm und Tastatur gibt es ein Problem. In der Regel erfolgen Ein- und Ausgabe gepuffert, d.h. erst beim Newline-Zeichen wird wirklich ein- oder ausgegeben. Damit ist das Auslösen von Aktionen mit einem einzigen Tastendruck ebenso wenig möglich wie z. B. die Ausgabe eines Fortschrittsbalkens. Bei der Ausgabe lässt sich das Problem recht einfach lösen. Es genügt die Zeile

```
$| = 1;
```

um die Pufferung abzuschalten. Bei der Eingabe ist es kniffliger. Man muss das Terminal in den ungepufferten Modus schalten, was mithilfe des `stty`-Kommandos auf Systemebene erfolgen kann:

```
# Eine einzige Taste einlesen
sub inkey
  {
  my ($key, $dummy);
  $dummy = `stty raw < /dev/tty > /dev/tty 2>&1`;
  $key = getc(STDIN);
  $dummy = `stty cooked < /dev/tty > /dev/tty 2>&1`;
  return(substr($key,0,1));
  }
```

Anstelle von `stty cooked ...` kann man auch `stty sane ...` verwenden und statt der Backticks auch die Funktion `system()`. Oder man holt sich ein Perl-Modul, in dem eine passende Funktion realisiert ist.

Binäre Daten und Dateien

In Perl werden Binärdateien ähnlich wie in C behandelt: Für das Lesen gibt es die Funktion `read`, fürs Schreiben `print`. Zum Positionieren dient `seek` und um die aktuelle Position im File zu erhalten, verwendet man `tell`. Diese Funktionen sind auch für Textdateien anwendbar, werden jedoch meistens in Dateien mit fester Record-Länge verwendet, wo ein wahlfreier Zugriff auf einzelne Sätze durch einfache Berechnungen möglich ist. Das folgende Beispiel zeigt das Kopieren einer Datei:

```
open FROM, "InFile";
open TO, ">OutFile";
while (read FROM, $buf, 16384)
  { print TO $buf; }
close FROM;
close TO;
```

Nehmen wir an, wir hätten eine Datei, die von einem C-Programm erzeugt wurde und zwar mit fester Satzlänge. Wir wollen diese Sätze auslesen und auf den Inhalt der einzelnen Felder zugreifen. Dazu verwenden wir die Funktionen `read` und `unpack`. `unpack` benötigt als Parameter ein Template, welches die Struktur des Satzes beschreibt, und einen String, welcher den Satz beinhaltet. Das Template ist eine Folge von Buchstaben, welche die Reihenfolge und Art der einzelnen Felder des Records beschreibt:

Templates für `pack` und `unpack`

A,a	ASCII String
b,B	Bit String
h,H	Hex String
c,C	Signed/Unsigned Char
s,S	Signed/Unsigned Short
i,I	Signed/Unsigned Integer
l,L	Signed/Unsigned Long
f	Float
d	Double
p	Pointer to a null-terminated string.

Weitere Templates sind in den Manual-Seiten zur Funktion `pack` zu finden. Ein Teil des C-Codes für obiges Beispiel sieht etwa folgendermaßen aus:

```
struct
  {
```

```
  char st[10];
  int in;
  double d;
  } drec;
  ...

  fwrite(&drec, sizeof(drec) 1 , fp);
  ...
```

Der Perl-Code zum Lesen der Datei lautet dann:

```
$template = \"a10 i d";
$len = length pack($templateo,\qq{,0,0);
open FP, "filename";
  ...
  read(FP,$rec,$len);
  ($str,$in,$d) = unpack($template,$rec);
  ...
```

Mit `pack` kann ein Record wie bei C erzeugt werden. Wir brauchen diese Funktion im obigen Beispiel nur, um die Länge des Records zu bestimmen.

Formatierte Ausgabe

Neben `print` kennt Perl auch die Ausgabefunktion `printf`, die wie bei der Programmiersprache C definiert ist. Erster Parameter von `printf` ist ein Format-String, in dem Platzhalter für die auszugebenden Variablen eingestreut sind. Solch eine Formatanweisung beginnt immer mit einem %-Zeichen. Danach folgt optional die Feldbreite und ein Buchstabe. Bei Gleitpunktzahlen kann die Anzahl der Ziffern vor und hinter dem Dezimalpunkt festgelegt werden (z. B. %10.2f). Strings lassen sich mit %s entweder linksbündig (Minus-Zeichen vor der Feldbreite) oder rechtsbündig (Plus-Zeichen vor der Feldbreite) einpassen. Die wichtigsten Formatanweisungen sind in der folgenden Tabelle zusammengefasst:

Formatanweisungen für `printf`	
%c	Zeichen (Char) im ASCII-Code
%s	String
%d	Zahl in Dezimaldarstellung
%U	Unsigned Integer, Dezimal
%O	Unsigned Integer, Oktal
%x	Unsigned Integer, Umwandlung in Hexadezimal
%X	Wie %x, nur `'A'-'F'` statt `'a'-'f'`
%e	Exponenten-Darstellung (z. B. 1.23456e-10)
%E	Wie %e
%f	Gleitkommazahl, Festpunktdarstellung
%g	Gleitkommazahl wie %e oder %f

Beispiel: Ausgabe in den verschiedenen Formaten. Die Feldbreite der Ausgabe ist zwölf Zeichen lang. Zwei Stellen der Gleitpunktzahl sollen als Nachkommastelle verwendet werden:

```
my $string = "Perlisch ist nicht schwer!";
my $int  42;
my $pi = 3.14159;

printf ("String linksbuendig      %-12s\n",$string);
printf ("String rechtsbuendig     % 12s\n",$string);
printf ("Integer linksbuendig     %-12d\n",$int);
printf ("Integer rechtsbuendig    % 12d\n",$int);
printf ("Gleitpunkt               %f\n",$pi);
printf ("Gleitpunkt als Exp.      %E\n",$pi);
printf ("Gleitpunkt linksbuendig  %-10.2f\n",$pi);
printf ("Gleitpunkt rechtsbuendig %10.2f\n",$pi);
```

Perl kann auch einfache Reports und Tabellen ausgeben. Man deklariert das Layout der Ausgabe mit `format` und gibt die einzelnen Records mit `write` aus. Die Formatdefinition kann irgendwo im Programm erfolgen und hat die folgende Syntax:

```
format FILEHANDLE =
fieldline_1
valueline_1
...

fieldline_n
valueline_n
.
```

Definiert ein Ausgabeformat für die Datei `FILEHANDLE`: `fieldline_i` definiert das Aussehen einer Zeile und valueline_i listet alle Werte und Variablen, die ausgegeben werden sollen. Zum Beispiel wird durch das Format die Ausgabe der Werte der Variablen $a, $b, $c und $d auf STDOUT definiert.

```
format STDOUT =
@#### @<<<<<  @|||||  @>>>>>
$a, $b, $c, $d
.
```

Dabei werden folgende Formatierungen vorgenommen:

$a wird durch die „#“-Reihe $a als Zahl in einem Feld der Länge 5 formatiert,

$b wird durch „<<<<<<“ als Zeichenkette linksbündig in einem Feld der Länge 6 formatiert,

$c wird durch die senkrechten Striche als Zeichenkette zentriert in einem Feld der Länge 6 formatiert und

$d schließlich $d als Zeichenkette rechtsbündig („>") in einem Feld der Länge 6 formatiert.

Mit `write FILEHANDLE;` schreibt Perl einen Datensatz entsprechend dem Format mit den aktuellen Werten der Variablen in die Ausgabedatei. `format FILEHANDLE_TOP = ...` definiert entsprechend ein Format für den Kopf einer Ausgabeseite respektive für `STDOUT` nur `TOP`.

Beispiel: Die Binärdatei von oben soll schön formatiert ausgegeben werden:

```
format TOP =
  STRING       INTEGER        DOUBLE
  ---------------------------------
  .

format =
  # String linksbuendig, Integer zentriert, Double rechtsbuendig
  @<<<<        @||||||        @>>>>>
  $str,$in,$d
  .

...

while (read(FP,$rec,$len))
 {
 ($str,$in,$d) = unpack($template,$rec);
 write;
 }

...
```

Verzeichnisse bearbeiten

Unter Perl lassen sich auch Verzeichnisse bearbeiten (ein komplexes Beispiel finden Sie in Form der lokalen Suchmaschine). Dazu wird ein Verzeichnis mit `opendir` geöffnet. Mit `readdir` lassen sich die Einträge lesen und mit `closedir` läßt sich das Verzeichnis wieder schließen:

```
opendir(DIR, "*") || die Oops!"
while ($eintrag = readdir(DIR))
 {
 print "$eintrag\n";
 }
closedir(DIR);
```

Das folgende Programm traversiert einen Dateibaum. Es wird ein Startverzeichnis angegeben und alle Dateien in diesem Verzeichnis werden aufgelistet. Danach

werden rekursiv alle im Startverzeichnis enthaltenen Unterverzeichnisse bearbeitet. Lediglich Dateien, die mit einem „." beginnen, werden ausgelassen. Dies sind zum einen die Verzeichnisse „." und „..", deren Bearbeitung zu endlosen Schleifen führen würde, und alle „versteckten" Dateien von Linux. Im folgenden Beispiel wird nur der Dateiname ausgegeben – es lassen sich natürlich die Dateien auch beliebig bearbeiten.

```
#!/usr/bin/perl
# perl script to traverse a file-tree
# File beginning with a '.' are omitted

my $path = $ARGV[0];

&scan_files($path);
print "\n";
exit(0);


sub scan_files
  {
  my (@scandirs,$scandir,@files,$file,$list);
  $scandir = @_[0];
  opendir(DIR,$scandir) || warn "can't open the directory $scandir: $!\n";
  @scandirs = grep {!(/^\./) && -d "$scandir/$_"} readdir(DIR);
  rewinddir(DIR);
  @files=grep {!(/^\./) && -f "$scandir/$_"} readdir(DIR);           ###
  closedir (DIR);
  for $list(0..$#scandirs)
    {
    &scan_files($scandir."/".$scandirs[$list]);
    }
  if ($#files > 0)
    {
    print "<b>$scandir</b> contains the following files:\n";
    foreach $file(@files)
      {
      print "$file\n";
      }
    }
  return 1;
  }
```

Wenn man in der durch `###` markierten Zeile den Ausdruck erweitert, kann auch die Menge der Dateien eingeschränkt werden, z. B. würde das Anhängen von `&& (/\.pm$|\.pl$/)` nur noch die Perl-Quellen und -Module berücksichtigen.

Eine weitere Möglichkeit, auf Verzeichnisse zuzugreifen, ist das „globbing". Man wendet die Metazeichenersetzung der Shell auf eine Verzeichnisangabe an und erhält eine Array mit den dazu passenden Dateinamen. Das folgende Unterpro-

gramm erzeugt eine Datei „index.html“ mit Links auf alle HTML-Dateien eines Verzeichnisses („$HTMLPATH“ enthält einen Dateipfad):

```
sub make_index
 {
 open (IND,">$INDEX") || die "Kann $INDEX nicht anlegen!\n";
 print IND "<HTML>\n";
 print IND "<head><title>Gesamtindex</title>\n";
 print IND "</head>\n";
 print IND "<body bgcolor=\"#ffffff\" text=\"#000000\" ";
 print IND "link=\"#0000ff\" vlink=\"#cc00cc\">\n";
 print IND "<H2 ALIGN=CENTER>überschrift</H2>\n";
 $searchpath = $HTMLPATH . "*.html";
 @files = glob($searchpath);
 foreach $eintrag1 (@files)
   {
   $eintrag1 =~ s!^.*/!!;
   if ($eintrag1 ne 'index.html')
     {
     $eintrag2 = $eintrag1;
     $eintrag2 =~ s/.html//;
     print IND "<A HREF=\"$eintrag1\">$eintrag2</A><BR>\n";
     }
   }
 print IND "</body>\n";
 print IND "</html>\n";
 close(IND);
 }
```

Mit dem folgenden Beispielprogramm sollen alle Prozesse eines bestimmten Users gelöscht werden. Das Programm kann nur mit root-Berechtigung laufen:

```
#!/usr/bin/perl
use strict;

if ($#ARGV != 0)
  {
  print "Usage: $0 username\n" ;
  exit(1);
  }
my $actuser = shift;
my $err = &killuser($actuser);
print "Code $err\n";
exit 0;

sub killuser # (user)
  {
  # Geht nur beim User root
  # Die folgende Zeile ist ggf. anzupassen
  my $pscommand = 'ps -axo pid,user'; # ps soll liefern: PID und USERNAME
```

```
  my $user = $_[0];
  my @entry = ();
  my $name = '';
  my $pid = 0;
  my $line = '';
  my $signal = '';

  # Das Loeschen geschieht in drei Abstufungen:
  # erst ein TERM-Signal, dann ein HUP-Signal,
  # und zum Schluss fuer den Rest ein KILL-Signal:
  foreach $signal ("TERM","HUP","KILL")
    {
    print "Sending $signal signal to all processes ...\n";
    # Die Ausgabe des ps-Kommandos zeilenweise lesen und bearbeiten:
    open(PS, "$pscommand |");
    @entry = grep(/$user/,<PS>);
    close PS;
    return(0) if ($#entry == -1);
    foreach $line (@entry)
      {
      chomp($line);        # Newline am Ende weg
      $line =~ s/  */ /g;  # Mehrfache Leerzeichen weg
      $line =~ s/^ *//g;   # Lerrzeichen am Zeilenanfang weg
      ($pid,$name) = split(/ /,$line);
      print "$pid ";
      kill($signal,$pid);
      }
    print "\n";
    }
  return(0);
  }
```

Das folgende Programmbeispiel zeigt die Mächtigkeit von Perl bei der Bearbeitung regulärer Ausdrücke. Es erzeugt eine Cross-Referenz aller Befehlsworte, Variablen und Unterprogramme eines Perl-Quelltextes (auch für ähnliche Sprachen wie C, C++ oder PHP einsetzbar). Es besteht aus zwei Unterprogrammen. Das erste, `process()`, erzeugt die Referenzliste und das zweite, `result()`, gibt die Liste im HTML-Format aus, so dass sie mit dem Browser gelesen und ausgedruckt werden kann. Zur Speicherung dient ein Hash, dessen Keys aus den gefundenen Wörtern gebildet werden. Der Inhalt besteht aus den Zeilennummern der Fundstellen, jeweils durch ein Leerzeichen getrennt. Ein- und Ausgabedatei werden beim Aufruf über die Kommandozeile übergeben.

```
#!/usr/bin/perl
# xref - Build a cross-reference index file for the specified packages.
use strict;

# Parameter von der Kommandozeile lesen,
# Abbruch falls keine angegeben wurden
```

```
die "Usage: xref <source file> <output file> ...\n" unless ($#ARGV >= 1);
my $source = shift  @ARGV;
my $outfile = shift @ARGV;

my %xref = ();

process($source);
result($outfile);

# Ergebnisse als HTML ausgeben
sub result
  {
  my $outfile = shift;
  my $old = '';
  my $num = '';
  my $key = '';
  my $first = 0;
  my @numlist = ();

  open(OUT, ">$outfile") || die "Failed to create $outfile: $!";
  print OUT qq~
  <HTML>
  <HEAD>
   <TITLE>Cross Reference Index</TITLE>
  </HEAD>
  <BODY>
  <H1>Cross Reference Index</H1>
  <DL>
  ~;

  foreach $key (sort keys %xref)
    { # Begriff ausgeben
    print OUT "\n<DT>$key<DD>\n";
    @numlist = split(/\s/, $xref{$key});
    $old = ''; $first = 1;
    foreach $num (sort {$a <=> $b} @numlist)
      { # sortierte Zeilennummern ausgeben
      if ($num ne $old)
        {
        $old = $num;
        if ($first) { print OUT "$num"; $first = 0; }
        else { print OUT ", $num"; }
        }
      }
    }
  print OUT qq~

  </DL>
  </BODY>
  </HTML>
```

```
  ~;
  close OUT;
  }

sub process
  {
  my $source = shift;
  my $line = '';
  my @match = ();
  my @words = ();
  my $word = '';
  my $lineno = 0;
  my $insidecomment = 0;

  open(SRC, $source) || die "Failed to open $source: $!\n";
  while ($line = <SRC>)
    {
    # Zeile bearbeiten (in Worte zerlegen)
    chomp($line);
    $lineno++;
    # Leerzeichen am Ende entfernen
    $line =~ s/\s*$//;
    # Sonderbehandlung von '$#': $#var --> $var
    $line =~ s/\$\#/\$/g;
    # Kommentare bearbeiten (Beginn mit '#')
    $line =~ s/\#.*$//;
    # Zeilenkommentare von php/c++ behandeln ('//')
    $line =~ s|//.*$||;
    # Ende und Beginn mehrzeiliger Kommentare bearb. (/*...*/)
    if ($insidecomment)
      {
      if ($line =~ m|\*/|)
        { $insidecomment = 0; }
      next;
      }
    if ($line =~ m|/\*|)
      {
      $insidecomment = 1 unless (m|/\*.*\*/|);
      next;
      }
    # Strings in Gaensefuesschen (") bearbeiten
    while (@match = $line =~ m|([^\"]*)(\"[^\"]*\")(.*)|)
      { $line = $match[0] . $match[2]; }
    # Strings in einfachen Anfuehrungszeichen (') bearbeiten
    while (@match = $line =~ m|([^\']*)(\'[^\']*\')(.*)|)
      { $line = $match[0] . $match[2]; }
    # Alle nicht benoetigten Zeichen entfernen
    $line =~ s/[^\$\%\@\#A-Za-z0-9]/ /g;
    # Mehrfache Leerzeichen durch eines ersetzen
    $line =~ s/ +/ /g;
```

```
    # Fuehrende Leerzeichen entfernen
    $line =~ s/^\s*//;
    # Ist noch was uebrig fuer die Inddizierung?
    next unless ($line);
    # Worte einzeln bearbeiten
    @words = split(/\s/, $line);
    foreach $word (@words)
      { $xref{$word} .= $lineno . " "  if ($word =~ /^[\$\%\@\#A-Za-z]/); }
    }
  close SRC;
  }
```

10.4.11 Referenzen

Bei Referenzen handelt es sich um einen Datentyp, der noch relativ neu ist, es gibt ihn erst seit Perl 5.003. Die Notwendigkeit ergab sich aus der Situation, dass es nicht möglich war, in Perl geschachtelte Arrays bzw. Hashes zu erzeugen. Wie bereits im Abschnitt über Arrays beschrieben, werden die Elemente eines Arrays 2 innerhalb eines anderen Arrays 1 so behandelt, als wären es Elemente von Array 1. Somit geht die Information verloren, welche Elemente zu Array 2 gehören.

Benötigt werden geschachtelte Arrays z. B. zur Erzeugung von Matrizen. Auch Perls Variante der objektorientierten Programmierung basiert intensiv auf dem Konzept der Referenzen. Mit Referenzen kann man also mehrdimensionale Arrays in Perl realisieren – oder Konstrukte, die noch weit komplexer sind. Referenzen sind immer Skalare, benötigen also den Dereferenzierer $ zu Beginn ihres Namens.

Normalerweise ist es nicht möglich, mehrere Arrays einem Array zuzuweisen. Mit Referenzen ist das Problem lösbar:

```
@a1 = ("Maier ","Huber ","Schulze ","Schmidt ");
@a2 = ("machen ","jeden ","Bloedsinn ","mit ");
(@Namen,@Spruch)=(@a1,@a2);
print @Namen;
```

Ein Anfänger würde erwarten, dass in das Array `@Namen` die Namen und in das Array `@Spruch` der Spruch eingelesen wird. Wir wissen jedoch, dass alles im Array `@Namen` landet und `@Spruch` leer bleibt. Referenzen können hier helfen:

```
@a1 = ("Eins ","zwei ","drei ","schon ","vorbei ");
$zeiger = \@a1;
print $zeiger;
```

Eine Referenz wird also mit dem Backslash gebildet. Nach der zweiten Zeile des Beispiels enthält der Skalar `$zeiger` eine Referenz auf die Variable `a1` bzw. auf

deren Adresse. Würde man die Referenz als Skalar ausgeben lassen, könnte das Ergebnis z. B. `ARRAY(0x876522c)` lauten.

Der Wert in der Klammer ist die Adresse. Davor steht der Datentyp (ARRAY, SCALAR, HASH etc.). Der Zeiger zeigt auf den Speicherplatz des Rechners, wo das Array abgespeichert ist. Um auf den Inhalt der referenzierten Variablen zugreifen zu können, muss diese wieder dereferenziert werden. Dies geschieht, indem der Referenz der Dereferenzierer der ursprünglichen Variablen vorangestellt wird.

```
@a1 = ("Eins ","zwei ","drei ","schon ","vorbei ");
$zeiger = \@a1;
print $$zeiger[0];
```

Als Ergebnis erhält man nun „Eins".

Um auf einzelne Elemente eines referenzierten Arrays oder Hashes zugreifen zu können, wird oft auch der Pfeiloperator verwendet (im Listing oben). Es gibt noch eine dritte Variante, die zum gleichen Ergebnis führt (im Listing unten):

```
@array = (1,2,3);
$arrayref = \@array;
print $arrayref->[0];

@array = (1,2,3);
$arrayref = \@array;
print ${$arrayref}[0];
```

Alles was man mit einem Array machen kann, kann man auch mit der Referenz auf das Array machen, wie im folgenden Beispiel (Will man auf das gesamte Array zugreifen, verwendet man `$zeiger`):

```
@a1 = ("Eins ","zwei ","drei ","schon ","vorbei ");
$zeiger = \@a1;
foreach (@$zeiger)
  { print "$_"; }
```

Mit Hashes, Variablen und Subroutinen läuft das Verfahren genauso wie bei den Arrays ab. Die Dereferenzierung erfolgt wie oben. Der letzte Teil des Listings zeigt, wie man auf den ganzen Hash über eine Referenz zugreift.

```
%ampel = ("oben"=>"rot",
          "mitte"=>"gelb",
          "unten"=>"gruen");

$zeiger=\%ampel;
print $zeiger;            # Ergebnis z.B: HASH(0x8745228)
print $$zeiger{'oben'};   # Ergebnis: rot

foreach (keys(%$zeiger))
```

```
  { print "Key: $_ , Value: $$zeiger{$_} \n"; }
```

Auch bei Hashes gibt es die schon bei Arrays beschriebenen Varianten des Zugriffs:

```
%ampel = ("oben"=>"rot","mitte"=>"gelb","unten"=>"gruen");
$zeiger=\%ampel;
print "$$zeiger{'oben'}\n";
print "$zeiger->{'mitte'}\n";
print "${$zeiger}{'unten'}\n";
```

Man kann zwei Typen von Referenzen unterscheiden: **harte Referenzen** und **symbolische Referenzen**. Harte Referenzen zeigen auf die Speicheradresse beliebiger Variablen. Um eine Referenz zu erzeugen, wird der Backslash-Operator verwendet. Zum Beispiel:

```
$variable = 5;
$referenz = \$variable;
print $$referenz;
```

Für Arrays und Hash-Variablen existiert eine weitere Methode, um harte Referenzen zu erzeugen:

```
$arrayref = [1, 2, 'a', 'b', 'c'];
$hashref = { 'Ort' => 'Muenchen', 'Name' => 'Meier'};
```

Die Besonderheit steckt hier in den Klammern, die jeweils die rechte Seite der Zuweisung umschließen. Im ersten Beispiel sind das die eckigen, im zweiten Fall die geschweiften Klammern. Durch diese Schreibweise wird eine Referenz auf ein anonymes Array bzw. Hash erzeugt. „Anonym“ bedeutet, dass kein expliziter Variablenname vergeben wird. Das Array bzw. Hash kann nur über seine Referenz angesprochen werden. Eine Anwendung für diesen Fall findet sich z. B. bei der Erzeugung von Objekten.

Im Gegensatz zur Adresse enthalten symbolische Referenzen nur den Namen beliebiger Variablen. Bei der Referenzierung wird der Name der Variablen ohne Datentypkennzeichen der Referenz zugewiesen. Die Dereferenzierung erfolgt wie bei den harten Referenzen.

```
$variable1 = 5;
$variablenname = "variable1";
print "$$variablenname\n";
```

Es lassen sich auch mehrdimensionale Arrays über Referenzen bilden. Im ersten `print`-Befehl wird „due“ ausgegeben. `$arr[1]` ist ein Zeiger auf ein Array, ebenso `$arr[0]` und `$arr[2]`. Vom angesprochenen Array mit dem Inhalt `("uno","due","tre")`, suchen wir das Element mit dem Index 1, also „due“. Zwei weitere Schreibweisen sind möglich (wie bereits oben beschrieben):

```
@arr = (
  ["eins","zwei","drei"],
  ["uno","due","tre"],
  ["One","two","three"]
  );
print $arr[1][1];
print ${$arr[1]}[1];
print $arr[1]->[1];
```

Mit Hashes sieht das beispielsweise folgendermaßen aus:

```
@arr = (
  {'Maier'=>'Kurt','Huber'=>'Erwin','Hofer'=>'Andreas'},
  {'Maier'=>'030-454567','Huber'=>'030-47301388','Hofer'=>'030-45345476'},
  {'Maier'=>'Ingenieur','Huber'=>'Politiker','Hofer'=>'Freiheitskaempfer'}
  );
print $arr[2]{'Hofer'};
```

Gäbe es den Hofer nur einmal, wäre es einfach, so überlegen Sie vielleicht, was ausgegeben wird. Es wird „Freiheitskaempfer“ ausgegeben. `$arr[2]` ist ein Zeiger auf das dritte Array, das die Berufe abspeichert. Davon wollen wir den Beruf mit dem Schlüssel „Hofer“ haben.

Komplizierter wird es, wenn man Arrays und Hashes mischt und dann später alle Werte innerhalb einer Schleife auslesen will. Da Arrays anders dereferenziert werden als Hashes, muss man herausfinden können, auf wwelches Objekt der Zeiger eigentlich zeigt. Dazu gibt es die Funktion ref:

```
@arr = (
   {"Goethe"=>"1749","Schiller"=>"1759","Mann"=>"1875"},
   ["Faust","Wallenstein","Buddenbrooks"],
   {"Goethe"=>"1833","Schiller"=>"1805","Mann"=>"1956"},
   ["Frankfurt","Ludwigsburg","Lübeck"]
   );
print "ref($arr[0]), ref($arr[1]), ref($arr[2]), ref($arr[3])\n";
```

Die Ausgabe-Anweisung liefert uns `HASH, ARRAY, HASH, ARRAY`. Beispielhaft soll im folgenden Beispiel einmal das ganze Array `@arr` ausgelesen werden:

```
@arr = (
   {"Goethe"=>"1749","Schiller"=>"1759","Mann"=>"1875"},
   ["Faust","Wallenstein","Buddenbrooks"],
   );
foreach $i(@arr)
  {
  if(ref($i) eq 'HASH')
    {
    foreach (keys(%$i))
      { print "$_ wurde ${$i}{$_} geboren.\n"; }
    }
```

```
  if(ref($i) eq 'ARRAY')
    {
    foreach (@$i)
      { print "$_ ist ein wichtiges Werk der Literatur.\n"; }
    }
  }
```

10.4.12 Der Perl-Debugger

Viele betrachten Debugger als Teufelszeug, andere, wie Linus Torvalds, brauchen einfach nie einen. Oft erweisen sie sich aber als letzte Rettung bei Denkblockaden. Perl hat seinen Debugger sogar eingebaut. Aber Vorsicht! Zu leichtfertig lässt sich schlecht durchdachter Code mit einem Debugger geradeziehen, was die Software schwer zu pflegen oder schlecht erweiterbar macht.

Manchmal verbirgt sich der Fehler aber (scheinbar) nicht im eigenen Code, sondern in einem Fremd-Modul. Manchmal ist aber auch die Dokumentation missverständlich und man hat nur die falsche Parameterversorgung programmiert; oder der Code des Vorgängers ist zu kompliziert, um seinen Ablauf durch Studieren der Listings zu verstehen. Perls Debugger kreist die Fehler recht schnell mittels Breakpoints, Actions und Watchpoints ein.

Das folgende Listing zeigt ein Programm zur Fakultätsberechnung, das irgendwie nicht so will – es endet nicht.

```
#!/usr/bin/perl
my $x = <>;
my $y = fak($x);

print "$x  $y  \n";
exit(0);

sub fak
  {
  my $p = shift;
  return ($p * fak($p - 1));
  }
```

Soll das Script im Debugger laufen, verwendet man einfach den Parameter `-d` beim Perl-Aufruf, also beispielsweise `perl -d fak.pl`. Das führt zu folgender Ausgabe:

```
Loading DB routines from perl5db.pl version 1
Emacs support available.

Enter h or `h h' for help.

main::(fak.pl:2): my $x = <>;
```

```
DB<1> n
5
main::(fak.pl:3): my $y = fak($x);
DB<1>
```

Das Debugger-Kommando „n“ (Next) führt die erste Zeile des Scripts aus und es kann der Wert 5 eingegeben werden (siehe Ende der obigen Ausgabe). Statt alle folgenden Zeilen mit „n“ vollständig auszuführen, wird das Kommando „s“ (Step) verwendet. Nun steigt der Debugger in die Funktion `fak` hinab (wir steppen gleich zweimal):

```
DB<1> s
main::fak(fak.pl:10):       my $p = shift;
DB<1> s
main::fak(fak.pl:11):       return ($p * fak($p - 1));
DB<1>
```

Das Kommando „l“ (List) verschafft einen Überblick über die nächsten Zeilen:

```
DB<1> l
11==>     return ($p * fak($p - 1));
12        }
13
```

Um im Code weiter nach unten zu gehen, ohne ihn auszuführen, genügt ein weiteres „l“-Kommando. Alternativ auch „l 70+20“, um z. B. 20 Zeilen ab Zeile 70 aufzulisten. Die nächste ausführbare Zeile wird durch `"==>"` angezeigt.

Die Eingabe eines Punktes führt die List-Anzeige wieder zum Ausgangspunkt zurück. Die Eingabe von „r“ (Return) weist den Debugger an, die aktuelle Funktion bis zum Ende auszuführen und anschließend im Hauptprogramm anzuhalten. Wir steppen aber noch etwas weiter und lassen uns mit dem „p“-Befehl den Inhalt der Variablen `$p` ausgeben.

```
DB<2> s
main::fak(fak.pl:11):       return ($p * fak($p - 1));
DB<2> p $p
3
DB<3> s
main::fak(fak.pl:10):       my $p = shift;
DB<3> s
main::fak(fak.pl:11):       return ($p * fak($p - 1));
DB<3> p $p
2
DB<4> s
main::fak(fak.pl:10):       my $p = shift;
DB<4> s
main::fak(fak.pl:11):       return ($p * fak($p - 1));
DB<4> s
main::fak(fak.pl:10):       my $p = shift;
```

```
DB<4> s
main::fak(fak.pl:11):       return ($p * fak($p - 1));
DB<4> p $p
0
DB<5> s
main::fak(fak.pl:10):       my $p = shift;
DB<5> s
main::fak(fak.pl:11):       return ($p * fak($p - 1));
DB<5>
```

Mit jeder Ausgabe-Anweisung erhöht sich auch die Nummer im Prompt. Mit „H“ lässt sich eine History-Liste ausgeben. Der Programmfehler wird oben auch schon sichtbar: Es gibt kein Abbruchkriterium! Man kann im Debugger auch Variablen neu definieren oder Variableninhalte ändern und natürlich wieder ausgeben. Bei Hashes ist der Ausgabebefehl „x“ günstiger:

```
DB<5> %h = ("eins" => "1", "zwei" => "2");

DB<6> p %h
zwei2eins1
DB<7> x %h
0  'zwei'
1  2
2  'eins'
3  1
DB<8>
```

Um zum Beispiel `$p` nochmals auszugeben, genügt ein Ausrufezeichen, gefolgt von der Nummer des History-Eintrags, z. B. `!2`. Nun soll das Programm geändert und nochmals durchgesteppt werden:

```
main::(fak.pl:2): my $x = <>;
DB<1> s
4
main::(fak.pl:3): my $y = fak($x);
DB<1> s
main::fak(fak.pl:10):       my $p = shift;
DB<1> s
main::fak(fak.pl:11):       return (1) if ($p == 0);
DB<1> s
main::fak(fak.pl:12):       return ($p * fak($p - 1));
DB<1> s
main::fak(fak.pl:10):       my $p = shift;
DB<1> s
   ...
DB<1> s
main::fak(fak.pl:11):       return (1) if ($p == 0);
DB<1> s
main::(fak.pl:5): print "$x  $y  \n";
```

```
DB<1>
4
  24
main::(fak.pl:6): exit(0);
DB<1> s
DB::fake::(/usr/lib/perl5db.pl:2084):
2084:     "Debugged program terminated.  Use 'q' to quit or 'R' to restart
```

Der Perl-Debugger kann aber noch mehr. Meist ist das Abarbeiten des Programms Befehl für Befehl zu langweilig bzw. zu langwierig. Deshalb nutzt man oft Breakpoints oder Watchpoints. Als Beispiel soll das folgende Programm dienen, das ein Verzeichnis rekursiv durchläuft:

```
#!/usr/bin/perl

my $path = $ARGV[0];
&scan_files($path);
print "\n";
exit(0);

sub scan_files
  {
  my (@scandirs,$scandir,@files,$file,$list);
  $scandir = $_[0];
  opendir(DIR,$scandir) || warn "can't open the directory $scandir: $!\n";
  @scandirs = grep {!(/^\./) && -d "$scandir/$_"} readdir(DIR);
  rewinddir(DIR);
  @files=grep {!(/^\./) && -f "$scandir/$_"} readdir(DIR);
  closedir (DIR);
  for $list(0..$#scandirs)
    { &scan_files($scandir."/".$scandirs[$list]); }
  if ($#files >= 0)
    {
    print "<b>$scandir</b> contains the following files:\n";
    foreach $file(@files)
      { print $scandir."/".$file, "\n"; }
    }
  return 1;
  }
```

Zuerst suchen wir uns die passende Stelle für den Breakpoint, beispielsweise die Zeile 21. Dann wird mit dem Befehl `b 21` der Breakpoint gesetzt und mit `c` (Continue) das Programm gestartet. Am Breakpoint hält die Programmausführung an und Sie können sich die Variableninhalte ausgeben.

```
main::(scandir.pl:5):   my $path = $ARGV[0];
DB<1> l 18-22
18:       @scandirs = grep {!(/^\./) && -d "$scandir/$_"} readdir(DIR);
19:       rewinddir(DIR);
```

```
20:       @files=grep {!(/^\./) && -f "$scandir/$_"} readdir(DIR);
21:       closedir (DIR);
22:       for $list(0..$#scandirs)
DB<2>
DB<2> b 21
DB<3> c
main::scan_files(scandir.pl:21):          closedir (DIR);
DB<3> p @files
wid-iec.htmlscandir.plfak.plfly1_54_gif_create.zip
plz.perl-1.6.prog.tar.gzplz.perl-1.6.data.tar.gz
DB<4> c
main::scan_files(scandir.pl:21):          closedir (DIR);
DB<4> p @files
format.cssomega.gifshow_ads468_60_g.gifads.html
DB<5> c
main::scan_files(scandir.pl:21):          closedir (DIR);
DB<5> p @files
DB<6> c
<b>./wid_iec</b> contains the following files:
./wid_iec/format.css
./wid_iec/omega.gif
<b>.</b> contains the following files:
./wid-iec.html
./scandir.pl
./test.pl
./fly1_54_gif_create.zip
./plz.perl-1.6.prog.tar.gz
./plz.perl-1.6.data.tar.gz

DB::fake::(/usr/lib/perl5db.pl:2084):
2084:     "Debugged program terminated.  Use 'q' to quit or 'R' to restart.";
```

Mit dem Befehl `c 21` hätten Sie das Setzen des Breakpoints und den Programmstart in einem Rutsch erledigen können. Dann wäre dort aber kein permanenter Breakpoint gesetzt worden, der sich später wieder verwenden lässt. Statt nun jedes Mal von Hand die Variablen zu listen, können Sie auch eine Aktion definieren:

```
main::(scandir.pl:5):   my $path = $ARGV[0];
DB<1> a 21 print "@files\n";
DB<2> b 21
DB<3> c
main::scan_files(scandir.pl:21):          closedir (DIR);
wid-iec.html scandir.pl fak.pl fly1_54_gif_create.zip
plz.perl-1.6.prog.tar.gz plz.perl-1.6.data.tar.gz
DB<3>
...
```

Man kann beim Breakpoint auch noch eine Aktion (meist eine Bedingung) hinzufügen, z. B.: `b 55 $code eq "geheim"`. An diesem Breakpoint hält der Debugger nur an, wenn die Variable `$code` den String `geheim` enthält. In diesem Fall stoppt das Programm am Breakpoint und der Debugger zeigt den Inhalt der Zeile 55 an.

Sie haben schon gesehen, dass man mit `p` Variableninhalte ausgeben kann – wenn auch manchmal nicht sehr übersichtlich. Deshalb kennt der Debugger noch andere Befehle. `X` gibt alle Variablen im aktuellen Modul aus. Da viele dieser Variablen perl-spezifisch sind (z. B. `@_, $_ , $1` usw.), kann die Liste ziemlich lang werden. `X` zusammen mit einem Such-String gibt alle Variablen aus, deren Name mit dem Suchbegriff übereinstimmen. Geben Sie nur den Namen selbst an und nicht die Präfixe ($, oder %). „X foo“ gibt beispielsweise die Werte aller Variablen aus, deren Name „foo“ lautet (`$foo, foo` und `%foo`).

Der Befehl `V` wird verwendet wie `X`. Zusätzlich kann jedoch ein optionaler Modulname angegeben werden, um die Variablen dieses Moduls auszugeben. Diese Eigenschaft ist allerdings erst von Belang, wenn Sie mit Modulen arbeiten. Der X-Befehl birgt das Problem, dass lokale Variablen innerhalb von Subroutinen nicht erkannt werden. Um die Werte lokaler Variablen auszugeben oder kleinere Perl-Fragmente auszuführen, an deren Ergebnis Sie interessiert sind, verwenden Sie den `x`-Befehl wie in folgendem Beispiel. Wenn Sie Arrays oder Hashes ausgeben, wird deren Inhalt angezeigt. Die Ausgabe von `X` und `V` ist etwas einfacher zu lesen als die von `x`, besonders im Falle von Hashes. So sieht die Verwendung von `X` und `x` aus:

```
DB<4> x $input
0  'Helle World'

DB<5> X %haschmich
%haschmich = (
   '1' => 'rot'
   '2' => 'gelb'
   '3' => 'gruen'
)
```

Tracing: Mit dem Kommando `t` wird der Trace-Modus ein- oder ausgeschaltet. Der Debugger zeigt dann jede durchlaufene Zeile an. Als Beispiel diene uns nun wieder das erste Programm (Fakultätsberechnung):

```
main::(fak.pl:2): my $x = <>;
DB<1> t
Trace = on
DB<1> c
3
main::(fak.pl:3): my $y = fak($x);
main::fak(fak.pl:10):       my $p = shift;
main::fak(fak.pl:11):       return (1) if ($p == 0);
main::fak(fak.pl:12):       return ($p * fak($p - 1));
```

```
main::fak(fak.pl:10):       my $p = shift;
main::fak(fak.pl:11):       return (1) if ($p == 0);
main::fak(fak.pl:12):       return ($p * fak($p - 1));
main::fak(fak.pl:10):       my $p = shift;
main::fak(fak.pl:11):       return (1) if ($p == 0);
main::fak(fak.pl:12):       return ($p * fak($p - 1));
main::fak(fak.pl:10):       my $p = shift;
main::fak(fak.pl:11):       return (1) if ($p == 0);
main::(fak.pl:5): print "$x  $y  \n";
3
  6
main::(fak.pl:6): exit(0);
DB::fake::(/usr/lib/perl5db.pl:2084):
2084:     "Debugged program terminated.  Use ‘q’ to quit or ‘R’ to restart.";
```

Zum Abschluss noch ein kleiner Trick, um ein Programm jede ausgeführte Zeile anzeigen zu lassen. Dieses Tracing lässt sich über die Environment-Variable `PERLDB_OPTS` einstellen, bevor der Aufruf des Debuggers erfolgt, z. B.:

```
PERLDB_OPTS="NonStop=1 AutoTrace=1 frame=2" perl -d Programm
```

Die `AutoTrace`-Option setzt den Debugger in den Tracing-Modus, indem er jede Quellzeile erst ausgibt, bevor er sie ausführt. Mit der `NonStop`-Option hält der Debugger weder am Anfang noch am Ende an. Mit `frame=2` kommen beim Eintritt und beim Verlassen von Unterfunktionen „entering ...“- und „exiting ...“-Meldungen dazu. Wer zusätzlich übergebene Parameter und Rückgabewerte von Unterfunktionen braucht, setzt `frame=4`.

Eine Einführung in den Gebrauch des Debuggers finden Sie in jeder neuen Perl-Distribution mit dem Kommando `perldoc perldebtut`. Die ausführliche Dokumentation steht in „perldebug“.

Debugger-Befehle

Option	Bedeutung
	Programmausführung steuern
n	Nächste Zeile ausführen, danach anhalten
s	Nächste Zeile starten, in Unterfunktion anhalten
r	Aktuelle Funktion fertig durchlaufen, dann stoppen
R	Zurück zum Start und noch einmal ausführen
	Variablen anzeigen
p	Wert ausgeben
V	Werte ausgeben
x	Dump
X	Werte ausgeben
	Source-Navigation
l	Vorwärts blättern
-	Rückwärts blättern
v	Code um aktuelle Zeile herum zeigen
.	Zurück zur aktuellen Zeile
f	In eine andere Source-Datei wechseln
	Erweiterte dynamische Navigation
c ZZ	Code bis zu dieser Zeile ausführen, dann stoppen
c FF	Code bis zur Funktion ausführen, in ihr anhalten
b ZZ	Breakpoint in Zeile setzen
b FF	Breakpoint in Funktion setzen
b ZZ/FF	Breakpoint mit Bedingung
a ZZ/FF AA	Actionpoint in Zeile/Funktion
w ZZ/FF	Watchpoint in Zeile/Funktion Variable
¡ CC	Pre-Prompt setzen
L	Breakpoints, Watchpoints, Actions anzeigen
B/A/W	Breakpoints, Watchpoints, Actions löschen
	ZZ = Zeile, FF = Funktion,
	AA = Aktion, CC = Command

In dieser Einführung sind bei weitem nicht alle Eigenschaften und Möglichkeiten von Perl beschrieben worden, so fehlen z. B. Module und die objektorientierten Eigenschaften ebenso wie die Anwendung diverser Module, etwa das GD-Modul für die Erstellung von Grafiken oder Perl-TK, mit dem man grafische Benutzeroberflächen programmieren kann. Zum weiteren Studium sind die diversen Perl-Bücher aus dem Verlag O'Reilly zu empfehlen.

10.5 Aufgaben

1. Schreiben Sie ein Perl-Programm, das Zahlen einliest und sie aufsummiert. Wird statt einer Zahl der Buchstabe „e“ eingegeben, berechnet das Programm den Mittelwert der Zahlen und gibt ihn aus.

2. Schreiben Sie ein Programm, das eine Zahl in Worten ausgibt (etwa für das Bedrucken eines Scheckformulars). Die Zahl soll dem Programm auf der Kommandozeile übergeben werden. Beispiel:

   ```
   user$  perl drucke 1234
   -eins-zwei-drei-vier-
   user$
   ```

   Verwenden Sie für die Zuordnung zwischen Ziffern und zugehörigem Zahlwort einen Hash.

3. Schreiben Sie ein Programm, das Text von der Standardeingabe einliest und die Häufigkeit der einzelnen Buchstaben ermittelt. Die Liste der Häufigkeiten soll dann alfabetisch ausgegeben werden. Hinweis: Eine Zeile in Buchstaben zerlegt man mit der Anweisung `@chars = split (//,$line);`.

4. Schreiben Sie ein Programm, das beliebig viele Zeilen Text von der Standardeingabe einliest und dien Text in umgekehrter Reihenfolge wieder ausgibt. Dabei sollen die Zeichen innerhalb der Zeile umgekehrt werden (`reverse`) und auch die letzte Zeile als erste ausgegeben werden (`push` und `pop`).

5. Erzeugen Sie eine Datei namens `gebtag`, die folgenden Text enthält:

   ```
   11.2.1947:Thomas Alva:Edison
   4.1.1643:Isaac:Newton
   14.3.1879:Albert:Einstein
   27.12.1571:Johannes:Kepler
   ```

   Ergänzen Sie die Datei mit einem Testdatensatz, der das heutige Datum trägt.

   Schreiben Sie nun ein Programm das nach der Püfung ob `gebtag` existiert, die Datei einliest und alle Personen heraussucht, die heute Geburtstag haben. Es sollen Vorname, Name und Geburtsjahr ausgegeben werden. Hinweis: Das aktuelle Datum erhalten Sie mit

   ```
   ($sek,$min,$std,$tag,$mon,$jahr) = localtime(time);
   $mon++; $jahr += 1900;
   ```

6. Schreiben Sie ein Programm für den Administrator, das alle Prozesse eine auf der Kommandozeile angegebenen Benutzers killt. Rufen Sie dazu das `ps`-Kommando als Pipe auf und verarbeiten Sie dessen Ausgabe zeilenweise, indem Sie die Prozesse des angegebenen Users heraussuchen. Dann

extrahieren Sie die Prozessnummer aus der Zeile. Perl kennt den Befehl `kill(Signal,Prozessnummer)` mit dessen Hilfe Sie die Prozesse löschen können. Führen Sie die Prozedur nacheinander mit den Signalen `TERM`, `HUP` und `KILL` aus.

Kapitel 11

Basiskonfiguration

Einführung

Dieses Kapitel beschäftigt sich mit einigen Aspekten der Linux-Konfiguration. Nach einigen einleitenden Informationen geht es um elementare Funktionen, die Steuerung von so genannten cron-jobs, die Paketverwaltung und den Systemstart sowie einige Dinge, die ein Administrator wissen sollte. Dieses Kapitel soll Ihnen einen Blick hinter die Kulissen und Abläufe von Linux ermöglichen. Das Ziel ist es, dass Sie verstehen, was wie wo gesteuert und voreingestellt wird. Etliches haben Sie ja schon erfahren, etwa die Einstellung der Konsole oder das Anzeigen und Löschen von Prozessen. An dieser Stelle sei nur nochmals auf die Wichtigkeit der Datei- und Verzeichnis-Zugriffsrechte hingewiesen.

Leider unterscheiden sich verschiedene Distributionen in manchen Konfigurationsdetails. In diesem Buch versuchen wir, den gemeinsamen Nenner möglichst vieler Linux-Systeme zu beschreiben. Dennoch kann es vorkommen, dass gerade bei Ihrer Distribution einzelne Details ein wenig anders gelöst sind. In solchen Fällen bleibt Ihnen ein Blick in die Originaldokumentation nicht erspart – nicht zuletzt, weil sich ja auch die einzelnen Distributionen von Version zu Version oft erheblich ändern. Auf kurz oder lang müssen Sie also in jedem Fall lernen, selbst die Manuals, Hilfeseiten etc. zu lesen und zu verstehen. Wenn Sie Linux selbst auf Ihren Rechner installiert haben, sind Sie selbst auch der Systemadministrator, also diejenige Person, die sich um die Konfiguration des Rechners kümmert. Solange es um die Linux-Grundfunktionen geht, kann das jeder.

Die meisten Distributionen bieten recht komfortable Konfigurationsprogramme an, die während und auch nach der Installation verwendet werden können: `drakconf` bei Mandriva, diverse `redhat-xxx`-Programme bei Red Hat bzw. Fedora, YaST bei Suse etc. Diese Werkzeuge sollten bei Konfigurationsproblemen immer die erste Wahl sein! Sie sind speziell für Ihre Distribution optimiert und nehmen Ihnen viel Arbeit und Mühe ab. Neben diesen distributionsspezi-

fischen Konfigurationsprogrammen gibt es auch solche, die von einzelnen Distributionen unabhängig sind. Der populärste Vertreter ist momentan Webmin (*http://www.webmin.com/*).

Ausgefeilte Konfigurationswerkzeuge mit schönen Benutzeroberflächen haben nur den Zweck, Ihnen die Mühe abzunehmen, die Linux-Konfigurationsdateien direkt zu verändern. Gerade für Linux-Einsteiger ist dies zweifellos praktisch. Es gibt aber eine ganze Reihe von Gründen, sich dennoch mit den Konfigurationsdateien – und damit mit den Interna von Linux – auseinander zu setzen:

- Die Konfigurationsdateien können Sie mit jedem beliebigen Texteditor verändern, auch dann, wenn Sie in einer Textkonsole oder über ein Netzwerk per `ssh` arbeiten. Viele Konfigurationsprogramme stehen dagegen nur im Grafikmodus und nur im lokalen Betrieb zur Verfügung.
- Sobald Sie einmal verstanden haben, wie die Konfiguration einer bestimmten Linux-Funktion erfolgt, können Sie dieses Wissen bei nahezu jeder anderen Linux-Distribution anwenden. Die Bedienung vieler Konfigurationswerkzeuge ist dagegen distributionsspezifisch.
- Nur durch die direkte Veränderung der Konfigurationsdateien können Sie alle Aspekte einer Systemfunktion steuern. Konfigurationswerkzeuge beschränken sich dagegen oft auf einige wenige Details.
- Konfigurationsdateien lassen sich leicht von einem Rechner zum anderen kopieren. Das kann eine Menge Zeit sparen, wenn Sie beispielsweise Linux auf einem anderen Rechner neu installieren etc.
- Je besser Sie verstehen, wie die Konfigurationsdateien aufgebaut sind und welche Steuerungsmöglichkeiten sie bieten, desto besser verstehen Sie Linux.

Fast alle Linux-Konfigurationsdateien befinden sich im `/etc`-Verzeichnis. Gruppen zusammengehöriger Konfigurationsdateien befinden sich oft in Unterverzeichnissen (z. B. alle Konfigurationsdateien für das Grafiksystem X im Verzeichnis `/etc/X11`). Eine besondere Bedeutung haben die Unterverzeichnisse `init.d` bzw. `rc.d`: Sie enthalten alle Dateien, die für den Init-V-Prozess verantwortlich sind. Wichtig ist schließlich das Unterverzeichnis `sysconfig` mit distributionsspezifischen Konfigurationsdateien.

Es ist übrigens eine gute Idee, eine Sicherheitskopie des gesamten `/etc`-Verzeichnisses anzulegen. Damit können Sie nach Änderungen jederzeit rasch feststellen, wie der ursprüngliche Zustand einer bestimmten Konfigurationsdatei war. Da die Konfigurationsdateien auch in der Regel Kommentarzeilen (eingeleitet durch #) gestatten, ist es auch eine ungeschriebene Regel, nicht einfach Zeilen in der Konfigurationsdatei zu ändern, sondern die jeweilige(n) Originalzeile(n) zu kopieren und als Kommentar in der Datei stehen zu lassen. So kann man bei einer Fehlkonfiguration schnell sehen, wie es ursprünglich mal gewesen ist.

Wenn Sie eine Konfigurationsdatei in Ihrer Distribution nicht finden, kann das mehrere Ursachen haben: Eine Möglichkeit besteht darin, dass die zugrunde liegenden Programmpakete gar nicht installiert sind. Machen Sie sich daher mit der Paketverwaltung Ihrer Distribution vertraut (siehe Kapitel 11.8 ab Seite 590). Es kann auch sein, dass die Datei zwar existiert, sich aber bei Ihrer Distribution an einem anderen Ort befindet. Verwenden Sie zur Suche die Kommandos `locate`, `find` oder `grep`.

Bei manchen Programmen werden Änderungen an den Konfigurationsdateien erst wirksam, wenn Sie das Programm neu starten bzw. es explizit dazu auffordern, die Konfigurationsdateien neu einzulesen. Hierfür müssen Sie zumeist eines der beiden folgenden Kommandos ausführen:

```
root#  /etc/init.d/funktionsname restart
root#  /etc/init.d/funktionsname reload
```

Im Gegensatz zu Windows ist es fast nie erforderlich, den Rechner neu zu starten. (Ausnahmen sind nur Veränderungen am Linux-Kernel sowie einige hardware-spezifische Einstellungen, die unmittelbar beim Systemstart durchgeführt werden können.) Näheres dazu finden Sie auf Seite 608.

11.1 Konfiguration der Textkonsole

Bei modernen Linux-Distributionen werden Sie nie mit der Textkonsole konfrontiert. Bereits der erste Login erfolgt im X Window System. Wenn die X-Konfiguration allerdings scheitert oder aus anderen Gründen kein grafisches System zur Verfügung steht, müssen Sie sich mit den Textkonsolen anfreunden.

Die zentrale Anlaufstelle für die Tastaturverwaltung ist das Programm `loadkeys`. Es lädt eine Tabelle, welche die Zuordnung zwischen Tastaturcodes und Zeichen steuert (für uns ist `de-latin1.map` die richtige Tabelle). Die Tabellen liegen je nach Distribution an verschiedenen Stellen, suchen Sie ggf. nach „keymaps“. Mittels `loadkeys de-latin1` kann die Tabelle geladen werden. Damit die jeweilige Tastaturtabelle beim Systemstart automatisch geladen wird, muss die Einstellung bleibend gespeichert werden. Bei den meisten Distributionen erfolgt die Konfiguration in der Datei `/etc/sysconfig/keyboard`, wobei aber das Format dieser Datei je nach Distribution variiert!

Einige Programme und Kommandos – etwa das Kommando `crontab` – starten zum Ansehen oder Bearbeiten von Dateien selbstständig einen Editor. Diese Programme nehmen dabei auf die Umgebungsvariablen *EDITOR* oder *VISUAL* Rücksicht. Wenn diese Variablen nicht voreingestellt sind, wird normalerweise der `vi` gestartet. Die Voreinstellung erfolgt in der Datei `/etc/profile` durch eine Zeile der Art `export EDITOR=/usr/bin/vi`.

11.2 Datum und Uhrzeit

Die meisten Distributionen verfügen über Tools zur Einstellung der Systemzeit und des Datums; dennoch ist es wissenswert, wie die Zeit intern verwaltet wird: Wegen der internationalen Vernetzung von Rechnern ist die Verwendung einer weltweit einheitlichen Uhrzeit erforderlich, nämlich GMT. Diese Abkürzung steht für *Greenwich Mean Time*. Für diese Zeit ist als zweite Abkürzung auch UTC üblich (*Universal Time, Coordinated*).

Wenn Sie eine Datei speichern, dann wird nicht die aktuelle Ortszeit gespeichert, sondern eine auf diesen internationalen Standard umgerechnete Zeit (In Wirklichkeit wird die Anzahl der Sekunden ausgerechnet, die zwischen dem 1.1.1970, 0:0 Uhr GMT und der aktuellen Zeit vergangen sind.). Wenn Sie die Datei anschließend mit `ls -l` ansehen, wird die Uhrzeit wieder auf die Ortszeit am Standort des Rechners zurückgerechnet. So ist es möglich festzustellen, welche Datei aktueller ist: eine um 18:00 Ortszeit in Bielefeld oder eine um 12:30 Ortszeit in New York gespeicherte Datei.

Beim Hochfahren des Betriebssystems wird mit dem Kommando `hwclock` die Uhrzeit aus der CMOS-Uhr Ihres Rechners gelesen. Die Konfigurationsdatei `/etc/sysconfig/clock` enthält Informationen darüber, ob die CMOS-Uhr die lokale Zeit oder die GMT-Zeit enthält und in welcher Zeitzone sich Ihr Rechner befindet. Mit diesen Informationen kann die CMOS-Uhrzeit in GMT umgerechnet werden. Linux verfügt also ab jetzt über die GMT-Zeit. (Statt die notorisch ungenaue CMOS-Uhr des Rechners auszuwerten, können Sie die Zeit auch von einem genaueren Zeit-Server im Internet oder im lokalen Netzwerk lesen). Für die Zurdnung der Zeitzonen wird von den Programmen in der Regel die Datei `/etc/timezone` ausgewertet. Sie ist eine Kopie einer Zeitzonendatei aus dem Verzeichnis `/usr/share/zoneinfo`. (`/etc/timezone` kann auch ein symbolischer Link auf eine Zeitzonendatei sein. Weitere Informationen finden Sie in den `man`-Seiten zu `tzselect`, `tzconfig` und `tzset`.

11.3 Spracheinstellung, Internationalisierung, Unicode

11.3.1 Zeichensätze

Ein Zeichensatz (*character set*) beschreibt die Zuordnung zwischen Zahlencodes und Buchstaben. Bekannte Zeichensätze sind ASCII (7 Bit), ISO-*n* (8 Bit) und Unicode (16 Bit).

- Der **ASCII**-Zeichensatz beschreibt lediglich 127 Zeichen, darunter die Buchstaben a-z bzw. A-Z, die Ziffern 0-9 sowie diverse Interpunktionszeichen.

- Die **ISO-8859**-Zeichensätze (Latin-Zeichensätze) enthalten neben den 127 ASCII-Zeichen bis zu 128 zusätzliche Sonderzeichen für verschiedene Sprachregionen. ISO-8859-1 (Latin-1) gilt beispielsweise für Westeuropa, ISO-8859-2 (Latin-2) für Zentral- und Osteuropa, ISO-8859-3 (Latin-3) für Südeuropa oder ISO-8859-4 (Latin-4) für Nordeuropa.

- Um das Durcheinander verschiedenster 8-Bit-Zeichensätze zu lösen, wurde der 16-Bit-Zeichensatz **Unicode** alias ISO-10646 entworfen. Damit können nicht nur alle europäischen Sonderzeichen kodiert werden, sondern darüber hinaus noch die meisten asiatischen Zeichen.

  Unicode regelt nur, welcher Code welchem Zeichen zugeordnet ist, nicht aber, wie die Codes gespeichert werden. Die einfachste Lösung scheint auf den ersten Blick darin zu bestehen, jedes Zeichen einfach durch zwei Byte (also 16 Bit) darzustellen. Diese Formatierung wird UTF-16 genannt (*Unicode Transfer Format*). Dabei verdoppelt sich allerdings der Speicherbedarf und es tritt der Bytecode 0 an beliebigen Stellen in Unicode-Zeichenketten auf. Viele C-Programme, E-Mail-Server etc. setzen aber voraus, dass das Byte 0 das Ende einer Zeichenkette markiert.

  Deswegen gibt es auch andere Möglichkeiten, Unicode-Texte zu repräsentieren. Die bei weitem populärste Alternative zu UTF-16 ist UTF-8. Dabei werden die US-ASCII-Zeichen (7 Bit) wie bisher durch ein Byte dargestellt, deren oberstes Bit 0 ist. Alle anderen Unicode-Zeichen werden durch zwei bis vier Byte lange Byte-Ketten dargestellt. Wegen der größeren Kompatibilität zu existierenden Programmen und einer Reihe anderer Vorteile hat sich UTF-8 unter Unix/Linux als Standard etabliert (während unter Microsoft-Windows häufig UTF-16 verwendet wird). Wenn im Zusammenhang mit Linux also von Unicode die Rede ist, ist in den meisten Fällen Unicode im UTF-8-Format gemeint.

Fast alle Linux-Anwendungen kommen mit allen Latin-Zeichensätzen zurecht, die meisten neueren Anwendungen sind zudem Unicode-kompatibel (diverse ältere Kommandos hingegen noch nicht). Per Default gilt mittlerweile bei fast allen Distributionen der Unicode-Zeichensatz (UTF-8). Der aktive Zeichensatz entscheidet auch darüber, wie Zeichen in Textdateien bzw. in Dateinamen kodiert werden. Die Dateisysteme von Linux kommen mit jedem Zeichensatz zurecht. Als Dateiname gilt jede Zeichenkette, die mit dem Byte-Code 0 endet. Je nachdem, welcher Zeichensatz gerade gültig ist, kann die Byte-Folge und -Anzahl für einen Dateinamen wie `äöü.txt` aber ganz unterschiedlich sein! Wenn der aktuelle Zeichensatz Latin-1 lautet, kann dieser Name durch sieben Byte (plus ein 0-Byte) ausgedrückt werden. Wenn als Zeichensatz dagegen Unicode/UTF-8 verwendet wird, ist der Dateiname zehn Byte lang (weil zur Darstellung von ä, ö und ü jeweils zwei Byte benötigt werden).

Probleme treten am häufigsten auf, wenn Sender und Empfänger beim Austausch von (Text-)Dateien einen unterschiedlichen Zeichensatz verwenden. Die

Lösung derartiger Probleme ist meist das Kommando `recode`. Damit können Sie eine Zeichensatzkonvertierung durchführen:

```
user$  recode latin1..u8 $¡$ latin1dat $¿$ utf8dat
user$  recode u8..latin1 $¡$ utf8dat $¿$ latin1dat
```

Wenn Sie den Zeichensatz für die Namen zahlreicher schon vorhandener Dateien ändern möchten, hilft das Kommando `convmv` weiter; zu finden ist es unter *http://j3e.de/linux/convmv/*.

11.3.2 Lokalisation und Zeichensatz einstellen

Die meisten Distributionen bieten Werkzeuge an, um die Lokalisation bzw. den Zeichensatz einzustellen. Die Lokalisation wird oft in der Datei `/etc/sysconfig/i18n` gespeichert (Variable `LANG`). Intern wird sowohl die Lokalisation als auch der Zeichensatz durch Umgebungsvariablen wie *LC_CTYPE* und *LANG* gesteuert. Die Lokalisation kann kategorieweise durchgeführt werden. Bei korrekter Konfiguration können Sie beispielsweise für Datums- und Zeitangaben das in Deutschland übliche Format verwenden, Fehlermeldungen aber dennoch in Englisch anzeigen. Die folgende Tabelle zählt die wichtigsten Variablen auf.

Wichtige Lokalisationsvariablen	
`LC_CTYPE`	bestimmt den Zeichensatz.
`LC_COLLATE`	bestimmt die Sortierordnung.
`LC_MESSAGES`	bestimmt die Darstellung von Nachrichten, Fehlermeldungen etc.
`LC_NUMERIC`	bestimmt die Darstellung von Zahlen.
`LC_TIME`	bestimmt die Darstellung von Datum und Uhrzeit.
`LC_MONETARY`	bestimmt die Darstellung von Geldbeträgen.
`LC_PAPER`	bestimmt die Papiergröße.
`LANG`	bestimmt den Default-Wert für alle nicht eingestellten. LC-Variablen.
`LC_ALL`	überschreibt alle individuellen LC-Einstellungen.

Natürlich berücksichtigt nicht jedes Programm alle Kategorien (und viele Programme ignorieren die *LC*-Variablen vollständig). Wenn einzelne Kategorien nicht eingestellt sind, verwenden Programme als Default-Wert C bzw. POSIX (Fehlermeldungen sind auf Englisch, Datum und Zeit werden im amerikanischen Format dargestellt etc.).

Anstatt alle hier aufgezählten Variablen einzeln einzustellen, können Sie einfach die Variable *LANG* einstellen. Damit wird für alle undefinierten Variablen der *LANG*-Default-Wert verwendet. (Einzig bei *LC_COLLATE* bleibt die Default-Einstellung POSIX.) Bei den meisten Distributionen erfolgt die gesamte Sprach-Einstellung über die *LANG*-Variable. Noch stärker als *LANG* wirkt *LC_ALL*.

Wenn diese Variable gesetzt wird, gilt für alle Kategorien diese Einstellung (ganz egal, wie die anderen *LC_*-Variablen oder wie *LANG* eingestellt ist).

Den aktuellen Zustand der Lokalisationseinstellung können Sie am einfachsten mit dem Kommando `locale` ermitteln. Dieses Kommando wertet auch *LANG* und `LC_ALL` aus und ermittelt daraus die resultierenden Einstellungen, z. B.:

```
user$ locale
LANG=de_DE
LC_CTYPE="de_DE"
LC_NUMERIC="de_DE"
LC_TIME="de_DE"
...
LC_ALL=
```

Eine Liste aller möglichen Einstellungen für die Variablen können Sie mit `locale -a` ermitteln. Üblicherweise wird die Schreibweise x_y verwendet, wobei x durch zwei Buchstaben die Sprache und y durch zwei Buchstaben das Land bezeichnet. Im deutschen Sprachraum sollten Sie *de_DE* verwenden. Für die englische Default-Einstellung ist die Kurzschreibweise *C* erlaubt. Beachten Sie, dass es bei KDE für jede Sprache ein eigenes Lokalisationspaket gibt. Eine Veränderung der Lokalisation wird nur wirksam, wenn auch das entsprechende Sprachpaket installiert ist. Hintergrundinformationen zum Thema *Locales and Internationalization* finden Sie in der `man`-Seite zum Kommando `locale` und im glibc-Manual. Zusammen mit der Lokalisation wird auch der Zeichensatz eingestellt. Der Zeichensatz folgt dem Ländercode nach einem Punkt, z. B. *de_DE.ISO-8859-1* oder *de_DE.utf8*.

11.4 Benutzer- und Gruppenverwaltung

Bei der Benutzerverwaltung geht es in erster Linie darum, wer auf welche Dateien zugreifen darf, wer welche Programme ausführen darf, wer auf welche Hardware-Komponenten (Device-Dateien) zugreifen darf etc. Unter Linux wird dazu eine Liste von Benutzern verwaltet. Außerdem ist jeder Benutzer mindestens einer, ggf. auch mehreren Gruppen zugeordnet.

Prinzipiell können Sie als `root` die Benutzerverwaltung weitgehend manuell durchführen, indem Sie die in diesem Abschnitt beschriebenen Dateien direkt ändern oder textorientierte Kommandos wie `useradd` anwenden. Komfortabler und weniger fehleranfällig ist es, die mit den meisten Distributionen mitgelieferten Werkzeuge zur Benutzer- und Gruppenverwaltung einzusetzen.

11.4.1 Benutzer

Unter Linux (und generell bei Unix-ähnlichen Systemen) gibt es zwei Typen von Benutzern:

- Der **Super-User** alias **Systemadministrator** alias **root** hat üblicherweise den Usernamen `root`. Wer sich als `root` anmeldet, besitzt uneingeschränkte Rechte: Sie darf alle Dateien ansehen, verändern, löschen, alle Programme ausführen etc. Derart viele Rechte sind nur zur Systemadministration erforderlich. Alle anderen Aufgaben sollten aus Sicherheitsgründen nicht als `root` ausgeführt werden!
- **Gewöhnliche Benutzer:** Diese Benutzer verwenden Linux, um damit zu arbeiten. Sie haben uneingeschränkten Zugriff auf ihre eigenen Dateien, aber nur eingeschränkten Zugriff auf den Rest des Systems. Als Login-Name wird in der Regel der Name des Anwenders verwendet.

  Es gibt noch eine Reihe von Benutzern, die nicht für die interaktive Arbeit am Computer vorgesehen sind, sondern zur Ausführung bestimmter Programme. So wird der Webserver Apache nicht vom Benutzer `root` ausgeführt, sondern von einem eigenen Benutzer, der beispielsweise `wwwrun` heißt. Diese Vorgehensweise dient einer möglichst hohen Systemsicherheit.

Die Liste aller Benutzer wird in der Datei `/etc/passwd` gespeichert. Dort werden für jeden Benutzer der Login-Name, die UID, die GID, der vollständige Name, das Heimatverzeichnis und die Shell gespeichert. Dabei gilt folgendes Format:

Login-Name:Passwort:UID:GID:Voller Name:Heimatverzeichnis:Login-Shell

- Der **Login-Name** sollte nur aus Kleinbuchstaben (US-ASCII-Buchstaben und Zahlen) bestehen und nicht länger als acht Zeichen sein.
- Die **UID**-Nummer (User Identification) dient zur internen Identifizierung des Benutzers. Die Nummer wird insbesondere als Eigentümer-Iinformation zu jeder Datei gespeichert. `root` hat immer die UID 0. Serverdienste und Dämonen laufen mit UID-Nummern zwischen 1 und 499. Für gewöhnliche Benutzer sind Nummern ab 500 vorgesehen.
- Die **GID**-Nummer (Group Identification) gibt an, zu welcher Gruppe der Anwender gehört. Mehr Details zu Gruppen folgen im nächsten Abschnitt.
- Das **Heimatverzeichnis** ist der Ort, an dem der Benutzer seine privaten Daten speichern kann. Bei gewöhnlichen Benutzern wird dazu üblicherweise der Pfad `/home/`*login-name* verwendet.

  Damit bei neuen Benutzern sofort sinnvolle Default-Einstellungen für die wichtigsten Programme vorliegen, sollten beim Anlegen eines neuen Benutzers alle Dateien aus `/etc/skel` in das neu erzeugte Heimatverzeichnis kopiert werden. Viele Programme zum Anlegen neuer Benutzer erledigen diesen Schritt automatisch. Der Inhalt von `/etc/skel` stellt damit die Ausgangseinstellung für jeden neuen Benutzer dar.

- Da unter Linux mehrere Shells zur Auswahl stehen, muss in der `passwd`-Datei angegeben werden, welche **Login-Shell** zum Einsatz kommen soll. (vollständiger Dateiname der Shell mit Pfad).

Früher wurde in der Datei auch das Passwort gespeichert (weshalb es das Feld noch gibt), das ist mittlerweile aber unüblich. Statt des Passworts steht hier nur das Zeichen „x“. Die verschlüsselten Passwortinformationen werden in der separaten Datei `/etc/shadow` gespeichert (siehe unten).

11.4.2 Gruppen

Der Sinn von Gruppen besteht darin, mehreren Benutzern den gemeinsamen Zugriff auf Dateien zu ermöglichen. Dazu wird jeder Benutzer einer primären Gruppe (*initial group*) zugeordnet. Außerdem kann ein Benutzer beliebig vielen weiteren Gruppen (*supplementary groups*) zugeordnet werden, also Mitglied mehrerer Gruppen sein.

Die Datei `/etc/group` enthält die Liste aller Gruppen. Es gilt folgendes Format *Gruppenname:Passwort:GID:Benutzerliste*.

Die Zuordnung zwischen Benutzer und Gruppe erfolgt auf zwei Weisen:

- Die primäre Gruppe eines Benutzers wird in `/etc/passwd` gespeichert.
- Die Zugehörigkeit zu weiteren Gruppen wird gespeichert, indem der Name des Benutzers bei der entsprechenden Gruppe in der Datei `/etc/group` an die Benutzerliste angehängt wird, z. B.: „`backup:x:34:johnny,jim,jack`“.

Bei den GID-Nummern ist 0 für `root` vorgesehen, 1 bis 99 für Systemdienste. GID 100 ist normalerweise für die Gruppe `users` reserviert. Das Passwortfeld bei den Gruppen wird fast nie verwendet.

Beim herkömmlichen Verfahren (das seit vielen Jahren unter Unix/Linux zum Einsatz kommt), verwenden alle gewöhnlichen Benutzer die primäre Gruppe `users`. Bei einigen Distributionen erhält jeder Benutzer seine eigene primäre Gruppe (Gruppen-Name = User-Name).

11.4.3 Passwörter

Linux-Passwörter dürfen nur aus ASCII-Zeichen bestehen (keine internationalen Sonderzeichen!). Aus Sicherheitsgründen sollten Passwörter sowohl Groß- als auch Kleinbuchstaben sowie mindestens eine Ziffer enthalten. Sie werden unter Linux verschlüsselt gespeichert; eine Rekonstruktion des Passworts ist nicht einfach möglich. Der dabei eingesetzte DES-Algorithmus berücksichtigt nur acht Zeichen, weswegen längere Passwörter momentan sinnlos sind. Das kann sich in Zukunft aber möglicherweise ändern.

So genannte Passwort-Cracker (also Programme, um Passwörter herauszufinden) beschreiten daher den umgekehrten Weg: Sie testen einfach für Millionen zufälliger Passwörter (meistens Wörter aus einem Wörterbuch in Kombination mit Zahlen), ob der verschlüsselte Code dafür passt. Da viele Anwender einfallslose Passwörter verwenden und Computer immer schneller werden, führt dieser Weg erschreckend oft zum Ziel.

Um potenziellen Angreifern das Leben zu erschweren, werden bei neueren Linux-Systemen die verschlüsselten Passwort-Codes nicht direkt in `/etc/passwd` gespeichert, sondern in der getrennten Datei `/etc/shadow`. Der Vorteil besteht darin, dass diese Datei nur von `root` gelesen werden kann und muss. (`/etc/passwd` und `/etc/group` müssen für alle Benutzer des Systems lesbar sein, weil sie elementare Verwaltungsinformationen enthalten. Dabei gilt das Format *User-Name:Passwort-Code:d1:d2:d3:d4:d5:d6:reserved*.

Die Felder *d1* bis *d6* sind optionale Zeitangaben: *d1* gibt an, wann das Passwort zum letzten Mal geändert wurde (in Tagen seit dem 1.1.1970). *d2* gibt an, in wie vielen Tagen das Passwort geändert werden darf. *d3* gibt an, in wie vielen Tagen das Passwort spätestens geändert werden muss, bevor es ungültig wird. (Details zu den Feldern erhalten Sie mit `man 5 shadow`.)

Die Default-Werte für *d1* bis *d3* erlauben, das Passwort jederzeit zu ändern und sorgen für unbeschränkte Gültigkeit. Bei den meisten Systembenutzern (`bin`, `daemon`, `lp` etc.) wird statt eines Passworts nur ein Stern gespeichert. Das bedeutet, dass es überhaupt kein gültiges Passwort gibt, ein Login also unmöglich ist.

Eine Vielzahl von Parametern zur internen Administration von Passwörtern und Logins befinden sich in der Datei `/etc/login.defs`. Dort können Sie alle wichtigen Voreinstellungen treffen, die dann beim Anlegen eines neuen Benutzers gelten.

Um Ihr eigenes Passwort zu ändern, führen Sie das Kommando `passwd` aus. Sie werden jetzt zuerst nach Ihrem alten Passwort gefragt und dann zweimal hintereinander aufgefordert, ein neues Passwort einzutippen. Die Eingabe wird nicht am Bildschirm angezeigt. Nur wenn beide Eingaben übereinstimmen, wird das neue Passwort akzeptiert. Ab jetzt müssen Sie bei jedem Einloggen das neue Passwort verwenden. Während normale Benutzer nur Ihr eigenes Passwort ändern können, darf `root` auch die Passwörter fremder Anwender verändern, indem beim Aufrunf von `passwd` der User-Name angegeben wird.

Tipp

Was tun, wenn Sie Ihr `root`-Passwort vergessen haben? In diesem Fall müssen Sie Ihren Rechner mit einer Installations- oder Rettungsdiskette bzw. mit einer Live-CD (z. B. Knoppix) booten. Legen Sie nun mit `mkdir` ein neues Verzeichnis (in der RAM-Disk des Installations- oder Rettungssystems) an und binden Sie dort jene Partition ein, auf der sich Ihr Linux-System befindet:

```
mount -t ext3 /dev/xxx /verz
```

Jetzt haben Sie Zugriff auf die Datei `/verz/etc/passwd` bzw. `/verz/etc/shadow` und können darin das `root`-Passwort löschen. Den dazu erforderlichen Editor müssen Sie mit der Pfadangabe `/verz/usr/bin` starten.

Wenn Sie andere Benutzer daran hindern möchten, auf die gerade beschriebene Weise das `root`-Passwort zu verändern, müssen Sie im BIOS Ihres Rechners alle Bootmedien außer der ersten Festplatte deaktivieren (damit nicht von einer Diskette oder CD gebootet werden kann) und das BIOS durch ein Passwort absichern. Dieses Passwort sollten Sie dann aber wirklich nicht vergessen . . . .

11.4.4 Manuelle Benutzer- und Gruppenverwaltung

Wenn Sie zur Benutzerverwaltung nicht die Konfigurationswerkzeuge Ihrer Distribution verwenden möchten, aber auch keine Lust haben, Dateien wie `/etc/passwd` und `/etc/groups` mit einem Editor zu bearbeiten (geht am schnellsten, birgt die meisten Gefahren), können Sie die im Folgenden beschriebenen Kommandos einsetzen (für tiefergehende Informationen befragen Sie die Manual-Seiten):

- Mit dem im vorigen Abschnitt beschriebenen Kommando `passwd` können Sie Ihr eigenes Passwort und als `root` auch fremde Passwörter ändern.
- Das Kommando `chsh` (change shell) ändert die Default-Shell für einen Benutzer.
- Das Kommando `groups` zeigt alle Gruppen an, denen der aktuelle oder ein explizit angegebener Benutzer angehört.
- Das Kommando `newgrp` erzeugt keine neue Gruppe, sondern bestimmt die aktive Gruppe eines Benutzers, der mehreren Gruppen angehört. Nach dem Login gilt automatisch die primäre Gruppe als aktiv. Neu erzeugte Dateien gehören daher dieser Gruppe an. Mit `newgrp` kann eine andere Gruppe zur aktiven Gruppe gemacht werden.
- Mit den textorientierten Kommandos `useradd`, `usermod` und `userdel` kann ein neuer Benutzer-Account erzeugt, ein Account verändert und wieder

gelöscht werden. Im Anschluss an `useradd` muss mit `passwd` noch ein Start-Passwort definiert werden, sonst kann der neue Login nicht verwendet werden, z. B.:

```
root#  useradd -m johnny
root#  passwd johnny
New passwd: xxx
Re-enter new passwd: xxx
```

`usermod` kann dazu verwendet werden, einzelne Felder im User-Eintrag zu verändern oder um einen Account zu sperren (`usermod -L jack` sperrt den Account von Jack, `usermod -U jack` gibt ihn wieder frei. Auch die Gruppenzugehörigkeit und anderes lässt sich ändern.

- Mit den Kommandos `groupadd`, `groupmod` und `groupdel` erzeugen Sie eine neue Gruppe, ändern deren Eigenschaften und löschen die Gruppe wieder.

- Zur Erinnerung: Die Kommandos `chown`, `chgrp`, `chmod` ändern den Besitzer, die Gruppenzugehörigkeit und die Zugriffsbits einer Datei.

- Je nach der Einstellung von `/etc/login.defs` werden alle fehlerhaften Login-Versuche in `/var/log/faillog` protokolliert. Mit dem Kommando `faillog -u name` können Sie feststellen, wie viele fehlerhafte Login-Versuche bei einem bestimmten Benutzer seit der letzten gültigen Anmeldung aufgetreten sind.

  Mit `faillog -u name -m max` kann eine Maximalanzahl fehlerhafter Login-Versuche für einen bestimmten Benutzer fixiert werden. Wird diese Zahl überschritten, wird der Login blockiert, bis *root* den Login durch das Kommando `faillog -u name -r` wieder erlaubt.

  Sie können die maximale Login-Anzahl durch `faillog -m max` auch generell festlegen. Allerdings sollten Sie dann immer auch `faillog -u root -m 0` ausführen, damit `root` von dieser Schutzmaßnahme ausgeschlossen ist.

11.5 Systemprozesse (Dämonen)

Als Dämonen (englisch daemons) werden Hintergrundprozesse zur Systemverwaltung bezeichnet. Diese Prozesse werden bereits während des Hochfahrens des Rechners im Rahmen des Init-V-Prozesses gestartet (Details zum Systemstart finden Sie auf Seite 604). Bei Windows werden solche Prozesse „Services“ genannt.

Neben diesen Serverdiensten gibt es eine Reihe weiterer Hintergrundprozesse, bei denen es sich aber nicht um richtige Programme handelt, sondern um Teilprozesse des Kernels. Sie erkennen diese Prozesse daran, dass `ps axu` ihre Namen in eckige Klammern stellt.

Einige wichtige Systemprozesse

`cron`	startet andere Programme zu vorgegebenen Zeiten
`cupsd`	Druckerspooler
`devfsd`	verwaltet das `/dev`-Dateisystem
`httpd`	Webserver (z. B. Apache)
`kdm`	KDE-Login-Manager (alternativ xdm oder gdm)
`klogd`	protokolliert Kernel-Meldungen
`lpd`	BSD-Druckerspooler
`sendmail`	Mail-Server zum Versenden von E-Mails
`sshd`	Secure-Shell-Server
`syslogd`	protokolliert Systemmeldungen
`xfs`	X-Font-Server
`xinetd`	startet andere Netzwerkdämonen
...	

11.6 Prozesse automatisch starten (crontab)

Wenn Ihr Rechner scheinbar unvermittelt damit beginnt, die Festplatte zu durchsuchen oder sonstwie ungefragt herumwerkelt, ist die Ursache fast immer der automatische Start von Prozessen durch den Dämon `cron`. Dieses Programm wird beim Rechnerstart durch den Init-V-Prozess automatisch gestartet. Es wird einmal pro Minute aktiv, analysiert alle `crontab`-Dateien und startet die dort angegebenen Programme. `cron` wird zum großen Teil für Wartungsarbeiten verwendet – um Logging-Dateien abzuschneiden, um temporäre Dateien zu löschen, um Verzeichnisse zu aktualisieren etc. Grundsätzlich ist er aber für alle Arbeiten zuständig, die regelmäßig immer wieder ausgeführt werden sollen.

Die globale Konfiguration von `cron` erfolgt durch die Datei `/etc/crontab`. Darüber hinaus dürfen Benutzer ihre eigenen `cron`-Jobs in den benutzerspezifischen Dateien `/var/spool/cron/tabs/`*user* mithilfe des Kommandos `crontab` definieren.

Das Recht der benutzerspezifischen `cron`-Steuerung kann mit den beiden Dateien `/var/spool/cron/allow` und `/var/spool/cron/deny` eingestellt werden (früher waren diese Dateien in `/etc` angesiedelt). Wenn `allow` existiert, dürfen nur die hier eingetragenen Benutzer `cron`-Kommandos ausführen. Wenn `deny` existiert, sind die hier eingetragenen Benutzer ausgeschlossen. Existiert keine dieser Dateien, hängt es von der Kompilation von `cron` ab, ob irgendwelche Benutzer außer `root` `cron` verwenden dürfen.

Die Datei `/etc/crontab` enthält zeilenweise Einträge für die auszuführenden Programme. Ihre Syntax sieht so aus:

```
min hour day month weekday user command
```

`min`	gibt an, in welcher Minute (0-59) das Programm ausgeführt werden soll.
`hour`	gibt die Stunde an (0-23).
`day`	gibt den Tag im Monat an (1-31).
`month`	gibt den Monat an (1-12).
`weekday`	gibt den Tag der Woche an (0-7, 0 und 7 bedeuten jeweils Sonntag).
`user`	gibt an, für welchen Benutzer das Kommando ausgeführt werden soll (meist `root`).
`command`	enthält schließlich das auszuführende Kommando.

Wenn in den ersten fünf Feldern statt einer Zahl ein * angegeben wird, wird dieses Feld ignoriert. `15 * * * *` bedeutet beispielsweise, dass das Kommando immer 15 Minuten nach der ganzen Stunde ausgeführt werden soll, in jeder Stunde, an jedem Tag, in jedem Monat, unabhängig vom Wochentag. `29 0 * * 6` bedeutet, dass das Kommando an jedem Samstag um 0:29 Uhr ausgeführt wird.

Für die Zeitfelder ist auch die Schreibweise */*n* erlaubt. Das bedeutet, dass das Kommando jede *n*-te Minute/Stunde etc. ausgeführt wird. `*/15 * * * *` würde also bedeuten, dass das Kommando viertelstündlich (*n*:00, *n*:15, *n*:30 und *n*:45) ausgeführt wird. Ebenso lassen sich Listen angeben, deren Werte durch Kommata getrennt sind. Für die viertelstündliche Ausführung hätten Sie auch `0,15,30,45` schreiben können.

Die Zeitangaben schreiben übrigens immer den frühesten Startzeitpunkt vor, einzelne `cron`-Jobs können auch mal etwas verzögert beginnen.

Die Dateien `/var/spool/cron/tabs/`*user* haben fast dasselbe Format wie `crontab`. Der einzige Unterschied besteht darin, dass die `user`-Spalte fehlt (die Jobs laufen immer unter der Kennung des jeweiligen Users ab).

Um die globale `cron`-Konfiguration zu verändern, können Sie `/etc/crontab` bzw. die Dateien in `/etc/cron*` direkt mit einem Editor bearbeiten. Wenn Sie dagegen benutzerspezifische `cron`-Einträge vornehmen möchten, sollten Sie dazu das Kommando `crontab -e` einsetzen. Als Editor wird der durch die Shell-Variable `EDITOR` spezifizierte Editor verwendet. `crontab -l` listet die aktuellen Einträge auf, `crontab -r` löscht alle Einträge. Mittels `crontab dateiname` kann eine vorbereitete Datei als `cron`-Tabelle übernommen werden. Die Manual-Seiten zu `cron` und `crontab` geben weitere Informationen.

Bei den meisten Distributionen enthält `/etc/crontab` lediglich einige wenige Einträge, die bewirken, dass einmal pro Stunde alle Script-Dateien in `/etc/cron.hourly/*` ausgeführt werden, einmal pro Tag die Script-Dateien in `/etc/cron.daily/*` etc. `/etc/crontab` könnte beispielsweise so aussehen:

```
# /etc/crontab bei Red Hat
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
```

```
MAILTO=root
HOME=/
17 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

Das Programm `run-parts` führ einfach alle ausführbaren Dateien (Scripts, Programme) innerhalb eines Verzeichnisses aus.

Dieses Konzept erleichtert die `cron`-Konfiguration erheblich. Für die meisten Fälle reicht es aus, einfach eine zusätzliche Script-Datei in eines der vier `/etc/cron.xxx`-Verzeichnisse zu kopieren, damit die darin angegebenen Kommandos ausgeführt werden.

Eine direkte Veränderung von `/etc/crontab` ist nur erforderlich, wenn Sie mit den vorgegebenen Zeitabständen nicht einverstanden sind. (Beachten Sie aber, dass Ihre Änderungen in `/etc/crontab` bei einem Distributions-Update meist überschrieben werden.)

11.7 Das /proc-Verzeichnis

Sie können eine Menge interner Kernel-Informationen sowie Daten zu allen gerade laufenden Prozessen aus den Dateien des `/proc`-Verzeichnis entnehmen. Unter anderem ist dort jedem Prozess ein eigenes Unterverzeichnis zugeordnet. Innerhalb des Prozessverzeichnisses befinden sich dann einige Dateien mit diversen Verwaltungsdaten (z. B. die zum Start verwendete Kommandozeile). Diese Daten werden von diversen Kommandos zur Prozessverwaltung (z. B. `top`, `ps` etc.) ausgewertet.

Die Dateien im `/proc`-Verzeichnis sind keine echten Dateien und beanspruchen daher auch keinen Platz auf der Festplatte. Das gilt auch für die scheinbar sehr große Datei `/proc/kcore`, die den Arbeitsspeicher abbildet. Vielmehr handelt es sich bei diesem Verzeichnis um ein virtuelles Dateisystem, das der Linux-Kernel zur Verfügung stellt.

Die meisten der `/proc`-Dateien liegen im Textformat vor. Um die Dateien zu lesen, müssen Sie unter Umständen `cat` statt `less` verwenden. (Manche `less`-Versionen kommen mit den virtuellen Dateien des `/proc`-Verzeichnisses nicht zurecht.)

Ein paar interessante /proc-Dateien

`/proc/`n`/*`	Informationen zum Prozess mit der PID=n
`/proc/bus/usb/*`	USB-Informationen
`/proc/bus/pci/*`	PCI-Informationen
`/proc/cmdline`	LILO/GRUB-Boot-Parameter
`/proc/cpuinfo`	CPU-Informationen
`/proc/devices`	Nummern von aktiven Devices
`/proc/ide/*`	IDE-Laufwerke und -Controller
`/proc/interrupts`	Nutzung der Interrupts
`/proc/iomem`	Nutzung des IO-Speichers
`/proc/ioports`	Nutzung der IO-Ports
`/proc/modules`	Aktive Module
`/proc/mounts`	Aktive Dateisysteme
`/proc/net/*`	Netzwerkzustand und -nutzung
`/proc/partitions`	Partitionen der Festplatten
`/proc/scsi/*`	SCSI-Laufwerke und -Controller
`/proc/sys/*`	System- und Kernel-Informationen
`/proc/uptime`	Zeit in Sekunden seit dem Rechnerstart
`/proc/version`	Kernel-Version

Über das `/proc/sys`-Verzeichnis können Sie sogar Kernel-Parameter im laufenden Betrieb verändern. Eine sehr ausführliche Beschreibung aller `/proc`-Dateien finden Sie im Buch zur Linux-Kernel-Architektur.

11.8 Software- und Paketverwaltung

Linux verwendet für seine Software-Installation ein Paketverwaltungssystem. Dies ist eine Datenbank, die Informationen über alle bereits installierten Programme und zugehörigen Dateien (Software-Pakete) enthält. Neue Programme werden durch die Kommandos des Paketverwaltungssystems installiert. Der Vorteil dieses Konzepts besteht darin, dass Abhängigkeiten und Konflikte zwischen den Software-Paketen berücksichtigt werden können. Wenn beispielsweise ein Programm A die Bibliothek B voraussetzt, lässt das Paketverwaltungssystem die Installation von A erst zu, nachdem B installiert worden ist.

Momentan kommen je nach Distribution zwei verschiedene Paketverwaltungssysteme zum Einsatz: Red Hat, Fedora, Mandriva, Suse sowie zahllose weitere Distributionen verwenden das von Red Hat entwickelte Paketformat RPM. Debian und alle davon abgeleiteten Distributionen nutzen dagegen das Paketformat DEB. Die meisten Distributionen stellen jeweils eine eigene Benutzeroberfläche zur Verfügung, die bei der Installation von Paketen hilft, die sich auf den Installations-CDs befinden. Weiterhin stellen viele Distributionen ein eigenes Werkzeug zur Verfügung, um die installierten Pakete zu aktualisieren, sobald es Updates gibt.

Die in diesem Kapitel vorgestellten Werkzeuge benötigen Sie nur, wenn Sie Linux-Pakete aus dem Internet installieren möchten, die sich nicht auf den mitgelieferten DVDs/CDs befinden, wenn Ihre Distribution keine eigenen Paketverwaltungswerkzeuge zur Verfügung stellt oder wenn Sie mit diesen Werkzeugen nicht zufrieden sind.

- **rpm:** Dieses Kommando ist für die Installation und Paketverwaltung bei allen RPM-basierten Distributionen zuständig. (Auch `redhat-config-packages`, `up2date` und YaST greifen darauf zurück.)
- **APT-RPM:** APT-RPM ist eine Ergänzung zu `rpm`, die bei der Installation von Paketen aus dem Internet hilft. Dabei werden automatisch alle Paketabhängigkeiten aufgelöst. APT-RPM kann auch dazu verwendet werden, sämtliche installierten Pakete mit einem einfachen Kommando zu aktualisieren.
- **APT** und **dpkg:** Bei Debian basiert die Paketverwaltung auf dem Kommando `dpkg`, das ein Gegenstück zu `rpm` ist. APT baut darauf auf und bietet wie APT-RPM mehr Komfort bei der Installation von Paketen aus dem Internet.
- **tar:** Dieses Kommando hilft dabei, eine Menge Dateien zu einem Archiv zusammenzufassen bzw. ein derartiges Archiv wieder auszupacken. Viele Software-Entwickler, die keine Lust dazu haben, RPM- oder DEB-Pakete zu bilden, stellen stattdessen einfache `tar`-Archive (Spitzname „tar-ball“) zur Verfügung. Deren Installation ist nicht viel schwieriger als die eines richtigen Pakets, führt aber an der Paketverwaltung Ihrer Distribution vorbei und kann daher Konflikte verursachen.

Hinweis

Die Pakete einer Linux-Distribution sind aufeinander abgestimmt. Das bedeutet, dass sie einheitliche Bibliotheken nutzen, mit demselben Compiler kompiliert wurden etc. Linux-Einsteiger sind deshalb gut beraten, nur Software-Pakete von den Installations-CDs unter Zuhilfenahme des Paketmanagers ihrer Distribution zu installieren.
Fortgeschrittene Linux-Anwender, die zusätzlich Pakete aus dem Internet nutzen möchten, sollten möglichst nur solche Pakete installieren, die speziell für ihre Distribution gedacht sind. Der Versich, andere Pakete zu installieren, kann an fehlenden Bibliotheken oder nicht erfüllten Paketabhängigkeiten scheitern. Gegebenenfalls sollte man alternativ die Quellpakete herunterladen und die Binaries neu generieren.

11.8.1 RPM (RPM Package Manager)

Das Kommando `rpm` hilft bei der Installation und Verwaltung von RPM-Paketen. Die folgende Liste zählt einige Funktionen von `rpm` auf. Das Programm hilft dabei,

- im Rahmen einer Installation automatisch Änderungen in schon vorhandenen Dateien durchzuführen (etwa in Script-Dateien),
- ein Programm durch eine aktuellere Version zu ersetzen (wobei von geänderten Dateien automatisch Updates erstellt werden),
- alle Dateien eines Programms wieder zu entfernen,
- sicherzustellen, dass vor der Installation eines Programms auch alle Voraussetzungen erfüllt sind (dass also alle erforderlichen Bibliotheken in der richtigen Versionsnummer zur Verfügung stehen),
- zu überprüfen, ob eine Datei seit der Installation des Pakets verändert wurde,
- festzustellen, zu welchem Paket eine bestimmte Datei gehört.

Die erforderlichen Verwaltungsinformationen befinden sich in jedem RPM-Paket. Bei der Installation werden diese Informationen in eine Datenbank eingetragen (Dateien im Verzeichnis `/var/lib/rpm`).

Die meisten RPM-Pakete werden in zwei Varianten zur Verfügung gestellt: Die eine Datei befindet sich üblicherweise im Verzeichnis `RPMS` und enthält die für die Installation des Pakets erforderlichen Binärdateien (daher Binärpaket). Die zweite Datei ist im Verzeichnis `SRPMS` untergebracht und enthält den Quellcode, der erforderlich war, um das Binärpaket zusammenzustellen (Quellpaket). Die Paketdatei enthält neben den zu installierenden Dateien zahlreiche Verwaltungsinformationen: eine kurze Paketbeschreibung, abermals Informationen über Versionsnummern, die Einordnung in die Gruppenhierarchie, Abhängigkeiten von anderen Paketen etc.

`rpm` verwaltet eine Datenbank mit Informationen über alle installierten Binärpakete. Diese Datenbank wird in sieben `*.rpm`-Dateien im Verzeichnis `/var/lib/rpm` gespeichert. Die Dateien der `rpm`-Datenbank dürfen auf keinen Fall verändert werden! Damit die Dateien aktuell sind, dürfen Pakete nicht einfach durch Löschen der Dateien, sondern müssen durch Deinstallieren (`rpm -e`) entfernt werden. In RPM-Paketen werden zwar Paketabhängigkeiten gespeichert, das Kommando `rpm` kann diese aber nicht automatisch auflösen und der Platzbedarf für die RPM-Datenbank ist relativ hoch. Die häufigste Anwendung ist das Installieren von Paketen:

rpm --install [optionen] dateiname
rpm --install [optionen] ftp://ftpserver/dateiname

`-i` oder `--install` installiert das angegebene Paket. Wenn es sich um ein Binärpaket handelt, wird es automatisch am vorgegebenen Ort im `root`-Verzeichnis installiert. Bei der zweiten Variante wird automatisch eine ftp-Verbindung hergestellt.

Wird zusätzlich die Option `--test` verwendet, führt `rpm` keine Veränderungen durch, sondern gibt nur aus, was bei einer tatsächlichen Installation passieren würde. `rpm` versucht automatisch die Integrität signierter Pakete zu überprüfen. Dazu muss aber der öffentliche Schlüssel der Signatur zur Verfügung stehen. Wenn `rpm` auf die Kontrolle verzichten soll, übergeben Sie die Option `--nosignature`. Der Installationsort kann durch die zusätzliche Option `--root verzeichnis` verändert werden.

Mit der Option `--nodeps` können Sie ein Paket selbst dann installieren, wenn `rpm` glaubt, dass diverse Abhängigkeiten nicht erfüllt sind. Die Option `--force` erzwingt eine Installation in jedem Fall (selbst dann, wenn `rpm` glaubt, dass das Paket bereits installiert ist). Mit der Option `--noScripts` erreichen Sie, dass auf die automatische Ausführung von Installationsprogrammen verzichtet wird. Handelt es sich beim angegebenen Paket um ein Quellpaket, werden der Programmcode und die Konfigurationsdateien in `/usr/src` installiert. Der Installationsort kann in diesem Fall durch eine Veränderung von `/etc/rpmrc` eingestellt werden.

Mit `-U` oder `--upgrade` können Sie ein bereits vorhandenes Binärpaket aktualisieren. Von den veränderten Konfigurationsdateien des bisherigen Pakets werden automatisch Backup-Dateien erzeugt, alle anderen Dateien desselben werden gelöscht. Mit der zusätzlichen Option `--oldpackage` können Sie ein neueres Paket durch ein älteres ersetzen. Auch beim Upgrade kann durch `--nodeps` erreicht werden, dass Paketabhängigkeiten ignoriert werden.

rpm --upgrade [optionen] dateiname
rpm --upgrade [optionen] ftp://ftpserver/dateiname

Mit `rpm -V` bzw. `rpm --verify` können Sie überprüfen, ob sich irgendwelche Dateien eines Pakets gegenüber der Originalinstallation verändert haben. Hat sich keine Datei geändert, gibt `rpm` nichts aus. Als Parameter wird nicht wie bei der Installation der gesamte Dateiname, sondern nur der Paketname angegeben (also etwa `abc` für die Paketdatei `abc-2.0.7-1.i386.rpm`).

Die Option `-e` bzw. `--erase` entfernt ein vorhandenes Paket. Wenn Sie zusätzlich die Option `--nodeps` verwenden, wird das Paket selbst dann entfernt, wenn andere Pakete davon abhängig sind.

Die Option `-q` bzw. `--query` bietet sehr viele unterschiedliche Anwendungsmöglichkeiten, die die Syntax etwas unübersichtlich machen. Die drei ersten Syntaxvarianten beziehen sich auf bereits installierte Pakete, mit der vierten Variante können Informationen zu einer noch nicht installierten `*.rpm`-Datei ermittelt werden.

rpm --query -a
rpm --query [optionen] packagename
rpm --query -f [optionen] datei
rpm --query -p [optionen] packagedatei

In der einfachsten Form mit der zusätzlichen Option `-a` liefert `rpm` eine unsortierte Liste aller Pakete. Wenn Sie als Parameter einen Paketnamen angeben (etwa `abc`), testet `rpm`, ob das Paket installiert ist. Wenn das der Fall ist, liefert `rpm` den vollständigen Namen mit Versions- und Release-Nummer. Mit der zusätzlichen Option `-f` können Sie feststellen, zu welchem Paket eine bestimmte Datei Ihres Systems gehört. Das ist besonders bei Konfigurationsdateien sehr nützlich. Die Zusatzoption `-p` ermöglicht es, Informationen über ein noch nicht installiertes Paket zu ermitteln. Als Parameter muss in diesem Fall der Dateiname einer `*.rpm`-Datei verwendet werden. Diese Option kann auch auf Pakete mit Quellcode angewandt werden.

Die drei letzten Syntaxvarianten können mit einigen weiteren Optionen verbunden werden. `rpm` liefert dann nicht nur den Paketnamen mit Versions- und Release-Nummer, sondern auch weitere Informationen:

`-i` :	Kurzbeschreibung des Pakets, komprimierte Größe
`-l` :	Liste aller Dateien des Pakets
`-lv` :	wie oben, aber mit Dateigröße, Zugriffsrechten etc.
`-c` :	Liste aller Konfigurationsdateien des Pakets
`-d` :	Liste aller Dateien mit Online-Dokumentation zum Paket
`--Scripts`:	Liste aller Installationsprogramme

Die folgenden Query-Optionen dienen zur Feststellung von Paketabhängigkeiten. Mit den Optionen `whatprovides` und `whatrequires` können alle bereits installierten Pakete gesucht werden, die das Paket oder die Library `name` zur Verfügung stellen bzw. erfordern. Dabei ist auf die exakte Schreibweise zu achten.

rpm --query [optionen] --whatprovides name
rpm --query [optionen] --whatrequires name
rpm --query [optionen] --provides [-p] paketdatei
rpm --query [optionen] --requires [-p] paketdatei

Mit den Optionen `provides` und `requires` kann eine Liste aller Pakete und Libraries erstellt werden, die ein bestimmtes Paket zur Verfügung stellt bzw. benötigt. Die Optionen können auch für noch nicht installierte Paketdateien (Option `-p`) eingesetzt werden. `provides` und `requires` stellen gewissermaßen die Umkehrung zu `whatprovides` und `whatrequires` dar. Im einen Fall wird nach Abhängigkeiten gesucht, die sich aus einem Paket ergeben, im anderen Fall nach Paketnamen, die diese Abhängigkeiten erfüllen. Alle vier Optionen können mit `-i`, `-l`, `-lv`, `-c` und `-d` kombiniert werden (siehe oben).

Um in einem (noch nicht installierten) Paket eine bestimmte Datei zu suchen, verwenden Sie das folgende Kommando:

`root#` **rpm -qpli /cdrom/RedHat/RPMS/* | grep Dateiname**

Um nach Paketabhängigkeiten zu suchen, kann das Kommando modifiziert werden. Bei der Suche nach `libvga` würden Sie mit dem o. a. Kommando keinen Erfolg haben. Die Suche nach dem erforderlichen Paket sieht so aus:

```
root# rpm -qpi --provides *.rpm | grep libvga
```

11.8.2 APT-RPM

Das Kommando `rpm` eignet sich dazu, um ein einzelnes Paket mit `rpm -i` zu installieren. Gut funktioniert das allerdings nur, wenn das Paket lokal auf der Festplatte oder auf einer CD vorliegt und wenn alle Paketabhängigkeiten erfüllt sind. Die Kommandos `apt-xxx` des Pakets APT-RPM bieten darüber hinaus eine Reihe weiterer Funktionen:

- Als Datenquelle können RPM-Archive im Internet dienen. APT-Archive müssen neben den eigentlichen RPM-Paketen auch Metadaten zur Verfügung stellen, die Informationen über den Inhalt und die Abhängigkeiten aller Pakete enthalten.
- APT-RPM löst automatisch alle Paketabhängigkeiten auf, lädt alle erforderlichen Pakete und installiert sie.
- APT-RPM kann alle bereits installierten Pakete mit einem einzigen Kommando aktualisieren. Dazu wird für jedes Paket getestet, ob im Internet bereits eine neuere Version zur Verfügung steht und gegebenenfalls werden die entsprechenden Pakete heruntergeladen und einschließlich Abhängigkeiten installiert.

APT-RPM ist zudem nur zu empfehlen, wenn es zu Ihrer Distribution passende APT-RPM-Datenquellen gibt (Internet-Server, die RPM-Pakete samt APT-RPM-Metadaten zur Verfügung stellen). Suchen Sie gegebenenfalls mit einer Suchmaschine nach *apt rpm repository*. Vor allem müssen Sie (noch mit `rpm`) eine aktuelle Version von `apt-rpm` installieren. Download-Links für den Quellcode und für vorkompilierte Pakete finden Sie im Internet unter:

https://moin.conectiva.com.br/AptRpm
http://apt-rpm.tuxfamily.org/
http://www.fedora.us/wiki/FedoraHOWTO

Der zweite Schritt besteht darin, dass Sie APT-Datenquellen in `/etc/apt/sources.list` oder in eine beliebige Datei im Verzeichnis `/etc/apt/sources.d` eintragen. Jede Datenquelle wird durch die folgenden Informationen bezeichnet: RPM-Typ (*rpm* oder *rpm-src*), Server-Adresse (*http://name*) oder lokales Verzeichnis (*file:///pfad*), Paketverzeichnis und Paketgruppe(n) (*stable*, *updates*, *os*, *testing* etc.). Bitte verstehen Sie die folgenden Zeilen nur als Syntaxbeispiel! Genaue Angaben für APT-Datenquellen für diverse Distributionen finden Sie im Internet. Soweit vorhanden, sollten Sie einen geografisch nahe liegenden Mirror-Server verwenden.

```
# /etc/apt/sources.list oder /etc/apt/source.d/name.list
rpm     http://download.fedora.us/fedora/ fedora/1/i386 os updates stable
```

```
rpm-src http://download.fedora.us/fedora/ fedora/1/i386 os updates stable
# diverse Zusatzpakete
rpm     http://rpm.livna.org/ fedora/1/i386 stable
rpm-src http://rpm.livna.org/ fedora/1/i386 stable
```

Auf die *rpm-src*-Zeilen können Sie verzichten, wenn Sie an den Quellcode-Paketen nicht interessiert sind.

Nun stehen Ihnen diverse `apt`-Kommandos zur Verfügung, von denen wir im Folgenden nur die wichtigsten erwähnen. Bevor `apt-get` tatsächlich beginnt, Pakete zu installieren oder zu löschen, bittet es um eine Bestätigung.

- `apt-get update` aktualisiert die Paketlisten der in `sources.list` angegebenen Archive. Es werden aber keine Pakete heruntergeladen oder aktualisiert. Der einzige Zweck dieses Kommandos ist, dass `apt-get` weiß, welche Pakete im Internet zur Verfügung stehen. Das Kommando sollte vor jedem anderen `apt`-Kommando ausgeführt werden.
- `apt-get install name` sucht das Paket *name* auf allen APT-RPM-Servern, lädt es herunter und installiert es. Gegebenenfalls werden auch weitere Pakete geladen und installiert, um Paketabhängigkeiten zu erfüllen. Statt *name* können Sie auch den vollständigen Dateinamen einer lokalen RPM-Datei angeben. Damit wird diese RPM-Datei installiert, wobei ebenfalls Paketabhängigkeiten aufgelöst werden.
- `apt-get remove name` deinstalliert das Paket *name*.
- `apt-get check` überprüft, ob alle Paketabhängigkeiten erfüllt sind. Wenn das nicht der Fall ist, können Sie mit `apt-get -f install` eine Reparatur versuchen.
- `apt-get upgrade` testet für alle installierten Pakete, ob es neue Versionen gibt. Wenn das der Fall ist, werden diese Pakete installiert. Das Kommando aktualisiert also die gesamte durch APT-RPM verwaltete Software.
- `apt-get dist-upgrade` funktioniert wie *upgrade*, allerdings verwendet `apt-get` nun einen anderen Algorithmus zur Behebung von Paketkonflikten. Für die Distribution wichtige Pakete werden gegenüber weniger wichtigen Paketen bevorzugt.
- `apt-cache show name` zeigt Informationen zum Paket *name* an.
- `apt-cache search name` durchsucht die Paketdatenbank nach *name* und zeigt passende Informationen an.

Beispiel:

```
root#  apt-get update
...
```

```
Reading Package Lists... Done
Building Dependency Tree... Done

root# apt-get install abiword
apt-get install abiword
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  fribidi libwpd ots
The following NEW packages will be installed:
  abiword fribidi libwpd ots
0 upgraded, 4 newly installed, 0 removed and 15 not upgraded.
Need to get 4617kB of archives.
After unpacking 12.4MB of additional disk space will be used.
Do you want to continue? [Y/n]  Y
...
```

Wenn Sie sich nicht mit den `apt-get`- und `apt-cache`-Kommandos auseinander setzen möchten, können Sie die grafische Benutzeroberfläche `synaptic` verwenden. Zur Installation dieses Programms führen Sie einfach `apt-get install synaptic` aus.

11.8.3 DPKG und APT (Debian-Paketverwaltung)

Die Verwaltung von Debian-Paketen erfolgt auf zwei Ebenen: Auf der unteren Ebene ist das Kommando `dpkg` für die Installation und Verwaltung von Paketen verantwortlich. `dpkg` ist mit `rpm` vergleichbar. Das Kommmando kann weder Paketabhängigkeiten auflösen noch selbstständig Pakete aus dem Internet laden oder aktualisieren.

APT (*Advanced Package Tool*) baut auf `dpkg` auf. Es bietet dieselben Funktionen wie das auf Seite 595 bereits vorgestellte System APT-RPM. (Genau genommen ist APT-RPM eine Portierung, um die Vorzüge des Debian-APT-Systems auch für RPM-Pakete nutzen zu können.) Debian-Pakete werden in verschiedene Gruppen eingeteilt:

- *Main:* Das sind die Basispakete von Debian. Der Quellcode dieser Pakete ist unter einer Lizenz verfügbar, die den strengen Regeln des Debian-Projekts entspricht. Das garantiert, dass die Nutzung und Weitergabe wirklich frei im Sinne der Open-Source-Idee ist.
- *Contrib:* Pakete dieser Gruppe sind ebenfalls samt Quellcode frei verfügbar. Die Pakete können allerdings nur in Kombination mit *Non-Free*-Paketen verwendet werden. Das betrifft z. B. alle Programme, die auf Bibliotheken aufbauen, deren Lizenz in irgendeiner Weise Einschränkungen bedingt.
- *Non-Free:* Pakete dieser Gruppe sind zwar kostenlos verfügbar, ihre Lizenz entspricht aber nicht dem Open-Source-Ideal des Debian-Projekts.

- *Non-Us:* Pakete dieser Gruppe enthalten Software, die nicht aus den USA exportiert werden darf (z. B. Verschlüsselungsalgorithmen). Derartige Software wird meist außerhalb der USA entwickelt, um eventuellen Exportproblemem von vornherein aus dem Weg zu gehen.

Zudem unterscheidet Debian zwischen *stable*-, *testing*- und *unstable*-Paketen. Als *stable* gelten nur die Pakete, die Bestandteil der aktuellen, offiziellen Debian-Distribution sind. Leider gibt es nur recht selten neue Versionen der Debian-Distribution. Deswegen erhalten Sie nur dann ein einigermaßen aktuelles System, wenn Sie *testing*- und eventuell auch *unstable*-Pakete einsetzen. Allerdings gelten für diese Pakete nicht die relativ strikten Sicherheitskontrollen; im Fall von Sicherheitsproblemen kann es länger dauern, bis Updates zur Verfügung stehen.

DPKG

Dieser Abschnitt gibt einen knappen Überblick über die wichtigsten `dpkg`-Kommandos. In der Praxis werden Sie diese eher selten benötigen und stattdessen APT einsetzen.

`dpkg --install paketdatei` installiert das angegebene Paket. Wenn bereits eine ältere Version installiert ist, wird diese deinstalliert und durch die neue ersetzt. Während der Installation werden auch die im Paket vorgesehenen Konfigurations-Scripts ausgeführt. Vor der Installation wird sichergestellt, dass alle Paketabhängigkeiten erfüllt sind. Wenn das nicht der Fall ist, wird die Installation abgebrochen. Aus der Fehlermeldung geht in der Regel hervor, welche Pakete fehlen.

`dpkg --configure paketname` führt die Konfigurations-Scripts des angegebenen Pakets aus. In der Regel geschieht das bereits während der Installation; in manchen Fällen kann es aber notwendig sein, diesen Schritt explizit auszuführen.

Die beiden Kommandos `dpkg --remove paketname` und `dpkg --purge paketname` entfernen die angegebenen Pakete. Bei der zweiten Variante werden auch die Konfigurationsdateien gelöscht.

dpkg --list
dpkg --list `'muster'`

`dpkg --list` zeigt alle tatsächlich installierten Pakete an. Mit `dpkg --list 'muster'` werden auch alle nicht installierten Pakete berücksichtigt, die laut `/var/lib/dpkg/available` verfügbar sind.

Bei der Paketliste wird in der ersten Spalte ein Code aus zwei Buchstaben angegeben. Der erste Buchstabe gibt den gewünschten Status des Pakets an (i = installieren, n = nicht installieren, r/p = entfernen, h = halten), der zweite Buchstabe den tatsächlichen Status (i = installiert, n = nicht installiert, c = konfiguriert, u = entpackt, aber noch nicht konfiguriert, f = fehlgeschlagen).

Die Informationen stammen aus der Debian-Paketdatenbank, einer Sammlung von Dateien im Verzeichnis `/var/lib/dpkg`. Dort werden Meta-Informationen über alle installierten und verfügbaren Pakete gespeichert.

`dpkg --listfiles paketname` zeigt alle Dateien eines Pakets an. Das Kommando funktioniert nur für bereits installierte Pakete. Der Inhalt nicht installierter Pakete kann mit `dpkg-deb --contents dateiname` angezeigt werden.

Schließlich sucht `dpkg --search dateiname` nach dem Paket, aus dem die angegebene Datei stammt.

APT

APT ist ab Debian 3.0 das Standardsystem zur Paketverwaltung. Bei älteren Debian-Versionen muss das Programm explizit installiert werden. APT greift auf die `dpkg`-Kommandos zurück, setzt aber spezielle APT-Paketquellen voraus. Das sind Server, die neben den DEB-Paketen auch Meta-Informationen über den Inhalt der Pakete und deren Abhängigkeiten zur Verfügung stellen.

APT wird durch die Dateien in `/etc/apt` gesteuert. Normalerweise muss nur die Datei `sources.list` geändert werden. In diese Datei werden alle APT-Quellen eingetragen. Die folgenden Zeilen geben ein Beispiel für den Aufbau dieser Datei. Einzelne Zeilen sind hier nur aus Platzgründen mit \ auf zwei Zeilen verteilt.

```
# /etc/apt/sources.list
# stable
deb http://ftp2.de.debian.org/pub/debian stable main contrib non-free
deb http://ftp2.de.debian.org/pub/debian-non-US stable/non-US main \
  contrib non-free

# security updates for stable
deb http://security.debian.org stable/updates  main contrib non-free
deb http://security.debian.org testing/updates main contrib non-free

# testing
deb http://ftp2.de.debian.org/pub/debian testing main contrib non-free
deb http://ftp2.de.debian.org/pub/debian-non-US testing/non-US main \
  contrib non-free
```

Anstatt neue Datenquellen mit einem Editor in `sources.list` einzutragen, können Sie das Programm `apt-setup` zu Hilfe nehmen. Auch CDs können als APT-Datenquellen dienen. Die CDs können allerdings nicht einfach in `sources.list` eingetragen werden. Stattdessen müssen Sie für jede CD das Kommando `apt-cdrom add` ausführen.

Wenn Sie für ein bestimmtes Paket eine APT-Quelle suchen, verwenden Sie am besten die Suchmaschine *http://apt-get.org*. Im Suchergebnis sind auch die Zeilen enthalten, die Sie in `sources.list` eintragen müssen.

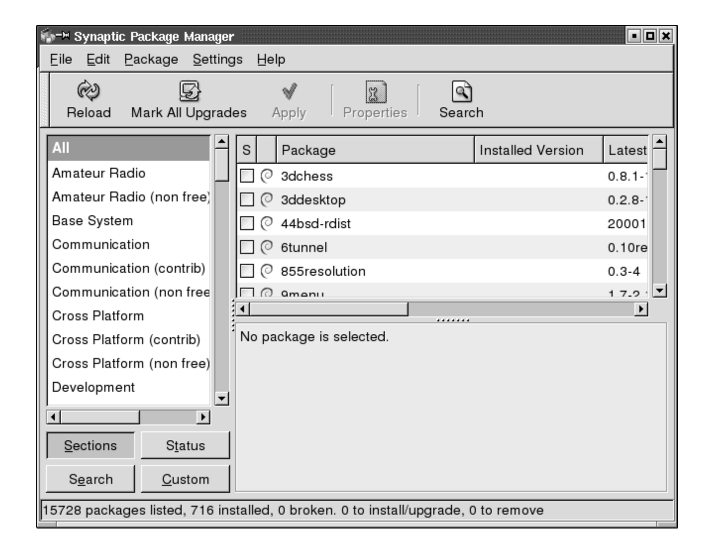


Abbildung 11.1: Auch für apt-get gibt es eine grafische Oberfläche: Synaptic.

Zur Installation und Administration von Paketen verwenden Sie die Kommandos `apt-get` und `apt-cache`. Die Syntax ist identisch mit der von APT-RPM (siehe Seite 596). Wie für APT-RPM steht auch für APT die Benutzeroberfläche `synaptic` zur Verfügung.

11.8.4 TAR

Vor allem bei erfahrenen Linux-Anwendern kommt es häufig vor, dass Linux-Software installiert werden soll, die nicht in Form eines Pakets einer bestimmten Distribution organisiert ist. Mit `gzip` komprimierte Archive weisen üblicherweise die Dateikennungen `*.tgz` oder `*.tar.gz` auf. Die Archive müssen mit dem Programm `tar` auf Ihrem Rechner installiert werden. Die Installation von Software-Paketen durch `tar` umgeht jedoch die Paketverwaltung des jeweiligen Systems. Die RPM- bzw. DEB-Datenbanken wissen daher nichts von den Programmen, die Sie mit `tar` installiert haben.

Befehl	Beschreibung
`root#` **tar -tzf archiv.tar.gz**	Inhalt des Archivs anzeigen
`root#` **tar -xzf archiv.tar.gz**	alle Dateien relativ zum aktuellen Verzeichnis installieren
`root#` **tar -xzf archiv.tar.gz "*.tex"**	nur *.tex-Dateien installieren
`root#` **tar -xzf archiv.tar.gz -C verz**	in ein Verzeichnis installieren

Immer häufiger wird zur Komprimierung der Archive das leistungsstärkere Programm `bzip2` verwendet. Sie erkennen derartige Archive an der Kennung

`*.tar.bz2`. Zum Anzeigen bzw. Dekomprimieren müssen Sie nun statt `-z` die `tar`-Option `-j` verwenden, also beispielsweise `tar -tjf archiv.tar.bz2`.

In vielen Fällen liegt die Software nur im Quellcode vor und muss vor ihrer Verwendung noch kompiliert werden. Das setzt voraus, dass Sie die wichtigen Entwicklungswerkzeuge (`gcc`, `make` etc.) sowie alle erforderlichen Bibliotheken (`devel`-Pakete) installiert haben.

11.9 Systemstart

In diesem Kapitel geht es um den Start eines Linux-Systems. Der Systemstart wird primär durch zwei Komponenten bestimmt:

- **GRUB** ist dafür verantwortlich, dass der Linux-Kernel gestartet wird. Dazu wird GRUB (ein winziges Programm) auf eine Diskette oder in den Bootsektor einer Festplatte installiert. (Als Alternative zu GRUB kann auch **LILO** eingesetzt werden.)
- Der **Init-V-Prozess** (das Programm `init`) kümmert sich darum, dass unmittelbar nach dem Kernel-Start verschiedene Initialisierungsarbeiten durchgeführt werden (Dateisysteme einbinden etc.) und diverse Systemprozesse gestartet werden (beispielsweise für alle Netzwerkfunktionen). Ein Grundverständnis dieses Prozesses hilft sehr bei der Durchführung von Konfigurationsarbeiten.

Neben diesen Schwerpunkten werden einige weitere Themen eher knapp angerissen:

11.9.1 GRUB

Genau genommen geht es in diesem Kapitel nicht um GRUB, sondern um GNU GRUB. GRUB (ohne GNU) bezeichnet eine alte, nicht mehr gewartete Version, von der GNU GRUB abgeleitet ist. Aus Gründen der Übersichtlichkeit verzichte ich aber darauf, GRUB ständig GNU vorzustellen.

Bevor die Einzelheiten der GRUB-Installation und -Konfiguration behandelt werden, ist es sinnvoll, sich ein erstes Bild davon zu machen, was während des Bootvorgangs passiert. Wenn Sie Ihren Computer einschalten, wird als Erstes das BIOS initialisiert. Während dieses Vorgangs erscheinen meist ein paar Systemmeldungen am Bildschirm, z. B. wie viel Speicher Ihr Computer hat.

Anschließend lädt das BIOS den Inhalt des ersten Sektors der ersten Festplatte oder der Diskette in den Speicher und führt diesen Code aus. Dieser spezielle Sektor der Festplatte heißt *Master Boot Record* (MBR).

Wenn auf dem Rechner nur Windows installiert ist, startet das im MBR befindliche Miniprogramm Windows. Bei Windows NT/2000/XP wird Windows nicht direkt gestartet, sondern zuerst der Windows-Boot-Loader. In dessen Menü können Sie dann unter Umständen zwischen mehreren Windows-Versionen auswählen.

Wenn auf dem Rechner auch Linux installiert ist, muss der MBR den Code eines speziellen Programms enthalten, das Linux startet. Es gibt mehrere **Linux-Boot-Loader**, wobei der populärste GRUB heißt (*GRand Unified Bootloader*). Der MBR ist nur 512 Byte groß. Aus diesem Grund ist es unmöglich, dass ein Boot-Loader vollständig im MBR Platz findet. Um das MBR-Platzlimit zu umgehen, enthält der MBR nur den Code, um den Rest des Boot-Loaders von der Festplatte zu laden.

Der GRUB-Code ist in zwei oder drei Teile zerlegt: *stage1* befindet sich im MBR und hat nur die Aufgabe, die ersten Sektoren von *stage1_5* oder *stage2* zu laden. *stage1_5* enthält Zusatzcode für den Zugriff auf Dateien in verschiedenen Dateisystemen. *stage2* enthält schließlich den eigentlichen Boot-Loader.

Sobald der Boot-Loader läuft, erscheint ein Menü mit einer Auswahl aller Betriebssysteme, die bei der GRUB-Konfiguration definiert wurden (meist Linux und Windows). Mit den Cursortasten können Sie nun das gewünschte Betriebssystem auswählen und dann mit ↵ starten. Oft ist der Boot-Loader so eingestellt, dass nach einer gewissen Zeit ein Betriebssystem automatisch gestartet wird.

Wenn Sie sich im Boot-Loader dafür entscheiden, Linux zu starten, muss der Boot-Loader die Linux-Kernel-Datei in den Speicher laden und starten. Die Kernel-Datei hat normalerweise den Dateinamen `/boot/vmlinuz`. (Der letzte Buchstabe *z* weist darauf hin, dass der Kernel komprimiert ist.) Der Boot-Loader muss also in der Lage sein, eine vollständige Datei aus einem Linux-Dateisystem zu laden.

An den Kernel werden meist einige Parameter übergeben, mindestens aber einer: der Device-Name der Root-Partition (z. B. *root=/dev/hdb3*). Damit weiß der Kernel, welches die Systempartition ist. Sobald der Kernel läuft, gibt er die Kontrolle an das Linux-Programm `/sbin/init` weiter. Dieses Programm ist für die Initialisierung des Linux-Systems zuständig.

Eine Besonderheit muss allerdings noch berücksichtigt werden: Der Linux-Kernel ist modularisiert. Das bedeutet, dass der Kernel an sich nur relativ elementare Funktionen enthält. Zusatzfunktionen zum Zugriff auf bestimmte Hardware-Komponenten, zum Lesen und Schreiben verschiedener Dateisysteme etc. befinden sich dagegen in Modulen, die bei Bedarf aus dem Dateisystem geladen werden und den Kernel so erweitern.

Damit der Startprozess gelingt, muss der Kernel aber auf die Systempartition zugreifen können. Falls diese Partition in einem Dateisystem vorliegt, das der Kernel nicht direkt unterstützt, oder wenn sich die Partition auf einer SCSI-Festplatte befindet, für die der Kernel keinen Hardware-Treiber enthält, tritt

ein Henne-Ei-Problem auf: Der Kernel kann nicht auf das Dateisystem zugreifen und daher die Module nicht laden, die er benötigen würde, um Dateien des Dateisystems zu lesen ....

Die Lösung des Problems besteht darin, dass der Boot-Loader nicht nur den Kernel lädt, sondern auch eine so genannte *initrd*-Datei. Dabei handelt es sich um eine spezielle Datei, die alle für den Startprozess erforderlichen Kernel-Module enthält. Die Datei steht dem Kernel vorübergehend als RAM-Disk zur Verfügung, d. h., der Kernel kann die erforderlichen Module unmittelbar nach dem Start von der RAM-Disk laden. (*initrd* ist die Abkürzung für *initial RAM disk*.)

- Wenn GRUB bereits installiert ist, erscheint unmittelbar nach dem Rechnerstart das GRUB-Menü. Sie können nun mit den Cursortasten ein Betriebssystem auswählen. Das ist aber noch nicht alles: Mit (Esc) können Sie das Menü verlassen, zusätzliche Startoptionen angeben etc.
- **Festplatten-Nomenklatur:** GRUB verwendet eine eigene Nomenklatur zur Bezeichnung von Festplatten bzw. Partitionen. Die Grundregel lautet, dass die Nummerierung immer mit Null beginnt. So ist beispielsweise `(hd0)` die erste Festplatte (`/dev/hda`) oder `(hd0,2)` die zweite Partition der ersten Festplatte (`/dev/hda2`).
- **GRUB-Konfigurationsdatei:** Das GRUB-Menü wird üblicherweise durch die Datei `/boot/grub/menu.lst` gesteuert.
- Die offizielle GRUB-Dokumentation können Sie mit `info grub` lesen. Als Ergänzung sind die FAQ auf der GRUB-Homepage, das Multiboot-with-GRUB-HOWTO sowie die beiden Artikel aus der Linux-Gazette bzw. Pro-Linux lesenswert:

  http://www.gnu.org/software/grub/
  http://www.tldp.org/HOWTO/Multiboot-with-GRUB.html
  http://www.linuxgazette.com/issue64/kohli.html
  http://www.pl-berichte.de/t_system/grub-howto.html

11.9.2 LILO

LILO (*Linux Loader*) war lange Zeit der wichtigste Boot-Loader für Linux. Seit GRUB aber von vielen Distributionen per Default eingesetzt wird, sinkt die Bedeutung von LILO. Da das Programm aber weiterhin auf vielen Linux-Systemen im Einsatz ist, erscheint hier zumindest eine knappe Beschreibung angebracht. Die Grundlagen des Bootprozesses, die im vorigen Abschnitt zu GRUB bereits behandelt wurden, werden hier vorausgesetzt. Generell empfehlen wir, LILO nur dann einzusetzen, wenn GRUB nicht verwendet werden kann.

11.9.3 Init-V-Prozess

Dieser Abschnitt beschreibt die Vorgänge, die beginnend mit dem Kernel-Start durch GRUB oder LILO bis zur Login-Aufforderung stattfinden. Nach dem Start des Kernels kann dieser vorläufig nur im Read-Only-Modus auf die Root-Partition zugreifen. Als erstes Programm startet der Kernel das Programm `/sbin/init`. Das Programm `init` kümmert sich dann um die Basiskonfiguration des Systems (Einbinden von Dateisystemen) und um den Start zahlloser Dämonen.

Wie das Programm `init` im Einzelnen ausgeführt wird, hängt von der jeweiligen Distribution ab. Zwar orientieren sich fast alle Distributionen am System-V-Init-Prozess, wie er auf vielen anderen Unix-Rechnern üblich ist. Dennoch gibt es Unterschiede: insbesondere in welchen Verzeichnissen sich welche Init-Dateien befinden, mit welchen Nummern oder Buchstaben die so genannten Runlevel bezeichnet sind, welche Konfigurationsdateien berücksichtigt werden etc.

Damit Sie nicht beim Lesen in den Details verloren gehen, folgt zunächst ein kurzer Überblick über einen normalen Linux-Systemstart:

- GRUB oder LILO lädt und startet den Kernel.
- Der Kernel startet das Programm `/sbin/init`.
- `init` wertet die Konfigurationsdatei `/etc/inittab` aus.
- `init` führt ein Script zur Systeminitialisierung aus.
- `init` führt das Script `/etc/rc.d/rc` oder `/etc/init.d/rc` aus.
- `rc` startet diverse Script-Dateien, die sich im Verzeichnis `/etc/rc.d/rc`*n*`.d` oder `/etc/init.d/rc`*n*`.d` befinden. (n ist der Runlevel – siehe unten.)
- Diese Script-Dateien starten verschiedene Systemdienste, insbesondere alle Dämonen für die Netzwerkfunktionen.

Der Kernel startet `/sbin/init` als erstes Programm. Dabei werden alle nicht ausgewerteten Kernel-Parameter weitergegeben. Auf diese Weise kann beispielsweise erreicht werden, dass Linux im Single-User-Modus gestartet wird. (Details finden Sie in der `man`-Page zu `init`.)

`init` ist also der erste laufende Prozess. Alle weiteren Prozesse werden entweder direkt von `init` oder indirekt durch Subprozesse von `init` gestartet. (Wenn Sie `pstree` ausführen, erkennen Sie sofort die dominierende Rolle von `init`.) Beim Herunterfahren des Rechners ist `init` der letzte noch laufende Prozess, der sich um das korrekte Beenden aller anderen Prozesse kümmert.

Für das Verständnis der System-V-Mechanismen ist der Begriff des Runlevels von zentraler Bedeutung: Sie können Ihren Rechner in unterschiedlichen Runlevels betreiben:

0: Herunterfahren des Rechners mit Halt
1 und **S**: Single-User
2: Multi-User ohne Netzwerk
3: Multi-User mit Netzwerk, aber ohne automatischen X-Start
5: Multi-User mit Netzwerk und automatischem X-Start
6: Herunterfahren des Rechners mit Reboot

Die Runlevel-1-Scripts werden ausgeführt, um von einem gewöhnlichen Runlevel (2, 3 oder 5) in den Single-User-Runlevel zu wechseln. Die Runlevel-S-Scripts kommen dagegen nur zur Anwendung, wenn der Single-User-Runlevel direkt nach dem Booten aktiviert werden soll (Kernel-Option *single*). Der Runlevel 4 hat üblicherweise keine Funktion.

Hinweis

> Die Runlevel-Nummerierung ist zwischen den Distributionen leider uneinheitlich. Insbesondere verwendet Debian andere Runlevel. Die jeweilige Bedeutung der Runlevel ist immer in `/etc/inittab` dokumentiert. Bei manchen Distributionen liefert `man init.d` ausführlichere Informationen.

Der Default-Runlevel wird durch die `initdefault`-Zeile in `/etc/inittab` bestimmt. Bei den meisten aktuellen Distributionen gilt 5 als Default-Runlevel. Der Default-Runlevel bestimmt, ob der Login nach dem Start des Rechners im Textmodus oder unter X erscheint. Um zwischen Text- und X-Login umzuschalten, ändern Sie einfach *n* in der Zeile `id:`*n*`:initdefault:`.

`root` kann den Runlevel auch im laufenden Betrieb durch das Kommando `init x` verändern. Für *x* muss eine Zahl oder ein Buchstabe angegeben werden, die bzw. der den Runlevel beschreibt. Beispielsweise ist es für manche Wartungsarbeiten sinnvoll, in den Single-User-Modus zu wechseln.

Inittab

Beim Systemstart wird `init` durch die Datei `/etc/inittab` gesteuert. Für die Syntax der `inittab`-Einträge gilt folgendes Schema:

```
id-code:runlevel:action:command
```

`id-code` besteht aus zwei Zeichen, die die Zeile eindeutig identifizieren. Der `runlevel` gibt an, für welchen Runlevel der Eintrag gilt. `action` enthält eine Anweisung für `init`. `command` gibt an, welches Linux-Kommando oder Programm gestartet werden soll.

Die folgende Liste zählt die wichtigsten `action`-Schlüsselwörter auf (eine vollständige Beschreibung erhalten Sie mit `man inittab`):

inittab-Schlüsselwörter

`ctrlaltdel`:	gibt an, wie `init` auf Strg+Alt+Entf reagieren soll.
`initdefault`:	definiert den Default-Runlevel für `init` (siehe oben).
`once`:	`init` startet das angeg. Kommando beim Runlevel-Wechsel.
`respawn`:	`init` startet das Kommando nach seinem Ende wieder neu.
`sysinit`:	`init` startet das Kommando einmal beim Bootprozess.
`wait`:	`init` wartet auf das Ende des nachfolgenden Kommandos.

Das folgende Listing gibt eine leicht gekürzte `inittab`-Datei wieder. Als Default-Runlevel ist 5 eingestellt. Bei einem normalen Systemstart führt `init` die Script-Dateien `rc.sysinit` und das Kommando `rc 5` aus. Außerdem wird der `update`-Dämon gestartet (alternativ: `bdflush`). Unabhängig vom Namen ist dies Programm dafür zuständig, dass gepufferte Datenblöcke regelmäßig auf der Festplatte gesichert werden. Schließlich wird für die Textkonsolen 1 bis 6 das Programm `mingetty` gestartet, das einen Login ermöglicht. (Wenn Sie mehr Textkonsolen haben möchten, ist hier der richtige Ort für Veränderungen.)

```
# Default-Runlevel
id:5:initdefault:

# Systeminitialisierung nach einem Reboot
# Dateisystem testen, Swapping aktivieren, Dateisysteme einbinden
si::sysinit:/etc/rc.d/rc.sysinit

# Start des jeweiligen Runlevel
# (Start von Netzwerkdiensten und anderen Systemprozessen (Dämonen))
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Reaktion auf <Strg>+<Alt>+<Entf>:
# (in 3 Sekunden Shutdown einleiten, Rechner dann neu starten)
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# gettys (Terminal-Emulatoren) für die Textkonsolen starten
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Terminal an der ersten seriellen Schnittstelle
T1:23:respawn:/sbin/getty -L ttyS0 9600 vt100
```

```
# Login ueber Modem auf der zweiten seriellen Schnittstelle
T2:23:respawn:/sbin/mgetty -x0 -s57600 ttyS1

# nur in Runlevel 5: xdm/kdm/gdm starten (X-Login-Manager)
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Die meisten Distributionen sind so vorkonfiguriert, dass die Tastenkombination (Strg) + (Alt) + (Entf) im Textmodus zu einem Neustart des Rechners führt. Wenn Sie möchten, dass der Rechner stattdessen ausgeschaltet wird, geben Sie beim `shutdown`-Kommando die Option `-h` statt `-r` an. Zum Deaktivieren der Tastenkombination setzen Sie vor die `ca::`-Zeile das Kommentarzeichen #.

Systeminitialisierung

Noch bevor die im nächsten Abschnitt beschriebenen rc-Dateien die runlevel-spezifischen Dienste starten oder stoppen, wird unmittelbar nach dem Rechnerstart eine Systeminitialisierung durchgeführt (*si:*-Zeile in `inittab`). Der Name des Scripts hängt von der Distribution ab; bei Red Hat bzw. Fedora ist es `/etc/rc.d/rc.sysinit` bei Suse `/etc/init.d/boot` und bei Debian sind es `/etc/rc.boot/*` sowie `/etc/init.d/bootmisc.sh`.

Während der Systeminitialisierung werden unter anderem folgende Dinge erledigt:

– diverse Systemvariablen initialisieren (inklusive Host- und Domainname)
– `/proc`-Dateisystem aktivieren
– eventuell Kernel-Optionen setzen
– Datum und Uhrzeit einstellen
– Tastaturlayout für die Textkonsole einstellen
– USB-Module laden, USB-Dateisystem aktivieren
– Dateisystem der root-Partition überprüfen (`fsck`)
– Root-Partition im Read-Write-Modus neu einbinden
– Swap-Partitionen einbinden
– eventuell RAID und LVM aktivieren
– Dateisystem der weiteren Partition überprüfen (`fsck`)
– Partitionen einbinden (ohne NFS-Partitionen)

Beachten Sie, dass nicht alle der hier beschriebenen Aufgaben direkt vom Systeminitialisierungs-Script durchgeführt werden. Zum Teil werden auch andere Script-Dateien verarbeitet. Dabei ist die Schreibweise `. name` üblich. (Der Punkt bewirkt, dass die angegebene Datei an dieser Stelle gelesen und ausgeführt wird. Anschließend wird das aufrufende Script fortgesetzt.)

Init-V-Scripts

Während der Systeminitialisierung werden die Dinge erledigt, die während des Rechnerstarts nur einmal getan werden (z. B. das Einbinden der Partitionen). Alle Netzwerkprogramme, die bei Bedarf (je nach Runlevel) gestartet werden sollen und die später (bei einem Runlevel-Wechsel) wieder beendet werden müssen, werden dagegen in zahllosen Init-V-Script-Dateien gestartet bzw. wieder gestoppt.

Die Init-V-Script-Dateien befinden sich je nach Distribution im Verzeichnis `/etc/init.d` oder in `/etc/rc.d/init.d`. Um eine höhere Kompatibilität zwischen den Distributionen zu erreichen, stellen meist Links sicher, dass beide Pfade gültig sind.

Zum Start der Init-V-Scripts führt `init` das Script `/etc/rc.d/rc` bzw. `/etc/init.d/rc` aus. An `rc` wird der gewünschte Runlevel *n* übergeben. `rc` führt zuerst einige Initialisierungsarbeiten durch. Dann werden alle `rc`*n*`.d/K`*nn*`*`-Script-Dateien zum Beenden laufender Prozesse ausgeführt (Kill). Schließlich werden alle `rc`*n*`.d/S`*nn*`*`-Script-Dateien zum Starten der neuen Prozesse für den jeweiligen Runlevel ausgeführt. (Wobei die wirklichen Scripts im Verzeichnis `/etc/init.d` stehen (ohne K*nN* bzw. S*nN* am Anfang) und in den `rc`*n*`.d/*`-Verzeichnissen nur Links auf diese Scripts. Das hat den Vorteil, dass die Script-Dateien für mehrere Runlevel verwendet und zentral verändert werden können. Zum Beispiel sind `K42xyz` und `S42xyz` Links auf das Script `xyz` in /etc/init.d. Die Zahlen legen fest, in welcher Reihenfolge die Scripts ausgeführt werden sollen (`s42xyz` kommt z. B. vor `s50abc`).

Der wesentliche Vorteil dieses auf ersten Blick unübersichtlichen Systems besteht darin, dass es sehr einfach ist, neue Systemprozesse in den Init-V-Prozess einzubauen: Es müssen lediglich das Script erstellt und die rc-Start- und Stopp-Links in die richtigen Verzeichnisse kopiert werden. (Genau das geschieht, wenn Sie mit `rpm` einen neuen Netzwerk-Server installieren.)

Hinweis

Ein Linux-Rechnerstart dauert relativ lange – selbst auf einem leistungsfähigen Rechner. Ein Grund dafür ist die rein sequenzielle Ausführung der Init-V-Scripts: Jedes Script kann erst gestartet werden, nachdem das vorige Script erfolgreich beendet wurde. Zur Beschleunigung dieses Prozesses gibt es Überlegungen, Init-V-Scripts parallel zu starten, soweit sie nicht voneinander abhängig sind. Sollte sich diese Idee durchsetzen, könnte sich der gesamte Init-V-Prozess in Zukunft fundamental verändern und deutlich von der Beschreibung in diesem Kapitel abweichen.

Viele Runlevel-Script-Dateien werden automatisch beim Rechnerstart bzw. bei einem Runlevel-Wechsel durch das *rc*-Script ausgeführt. Dabei wird der Parameter `start` oder `stop` übergeben, je nachdem, ob die betreffende Funktion gestartet oder beendet werden soll.

Ob ein automatischer Start/Stopp durchgeführt wird, hängt nur davon ab, ob es im `rcn.d`-Verzeichnis für den jeweiligen Runlevel einen Link auf das Init-V-Script gibt oder nicht. Wenn Sie also möchten, dass eine bestimmte Funktion in Zukunft automatisch aktiviert werden soll, müssen Sie derartige Links einrichten. Entsprechend müssen Sie die Links wieder entfernen, wenn Sie in Zukunft einen automatischen Start verhindern möchten.

Die folgenden Kommandos zeigen, welche Links Sie beispielsweise einrichten müssten, um das Programm `httpd` in Zukunft bei den Runlevels 3 und 5 automatisch zu starten.

```
root#  cd /etc/rc.d
root#  ln init.d/httpd rc0.d/K15httpd
root#  ln init.d/httpd rc1.d/K15httpd
root#  ln init.d/httpd rc2.d/K15httpd
root#  ln init.d/httpd rc3.d/S85httpd
root#  ln init.d/httpd rc5.d/S85httpd
root#  ln init.d/httpd rc6.d/K15httpd
```

Das Entfernen der Links verursacht weniger Tippaufwand:

```
root#  rm rc?.d/*httpd
```

In der Praxis werden Sie die obigen `ln`- bzw. `rm`-Kommandos selten manuell eintippen: Die meisten Distributionen stellen nämlich Kommandos zur Verfügung, die Ihnen diese Arbeit abnehmen, z. B. `insserv` bei Suse bzw. `chkconfig` bei Red Hat, Fedora und Mandriva.

Auch das Löschen der Links ist nicht unbedingt nötig. Wenn ein Dämon oder Dienst schon von der Installation her eingerichtet wurde oder Sie das automatische Starten nur temporär unterbinden wollen, genügt es, das Script in `/etc/init.d` umzubenennen (z. B. von `httpd` in `httpd.DISABLED`). Dann zeigen die Links ins Leere und es wird nichts gestartet.

Die Runlevel-Script-Dateien können auch manuell ausgeführt werden, z. B. so:

```
root#  /etc/init.d/httpd stop
```

Dadurch wird der Webserver Apache gestoppt. Bei vielen Distributionen sind zum Start von Dämonen auch eigene Kommandos vorgesehen, die weniger Tippaufwand bereiten. Beispielsweise hat `service httpd stop` unter Red Hat bzw. Fedora und Mandriva dieselbe Bedeutung wie das obige Kommando.

Neben „start“ und „stop“ haben die meisten Scripts noch weitere Parameter:

- `start` startet die betreffende Funktion.
- `stop` beendet die Funktion
- `status` zeigt eine kurze Information an, ob die Funktion aktiv ist oder nicht.

- `reload` bietet sich dann an, wenn geänderte Konfigurationsdateien neu eingelesen werden sollen, ohne den Dämon dabei ganz zu stoppen.
- `restart` bewirkt dagegen meistens, dass der Dämon vollkommen gestoppt und anschließend neu gestartet wird. Eventuell vorhandene Verbindungen zu Clients können dabei verloren gehen.

Bei manchen Scripts ist `reload` bzw. `restart` nicht vorgesehen. In diesem Fall müssen Sie das Script zuerst mit `stop` und nach einer kurzen Pause mit `start` ausführen.

11.10 Log-Dateien und Kernel-Meldungen

Linux protokolliert in so genannten Log-Dateien sehr viele Vorgänge: Kernel-Meldungen, das Ein- und Ausloggen von Benutzern, viele Netzwerkoperationen (z. B. die Herstellung einer Internet-Verbindung) etc. Die Log-Dateien befinden sich üblicherweise im Verzeichnis `/var/log`, manchmal auch in `/var/adm/log`. Log-Dateien sind sehr oft ein wichtiges Hilfsmittel zur Fehlersuche. Für das Log sind zwei Dämonen zuständig, die beide im Rahmen des Init-V-Prozesses gestartet werden: `klogd` ist für Kernel-Meldungen zuständig und `syslogd` protokolliert die Meldungen diverser Programme. Der Dämon wird durch die Datei `/etc/syslog.conf` gesteuert. In dieser Datei können Sie einstellen, was in welche Datei protokolliert werden soll.

Während des Linux-Starts huschen bei älteren Distributionen seitenweise Kernel- und Init-V-Meldungen über den Bildschirm. Bei moderneren Distributionen werden diese Meldungen meist hinter einer netten Grafik verborgen. Auch diese Meldungen werden protokolliert. In der ersten Phase des Startprozesses versucht der Linux-Kernel die Hardware zu erkennen. Die dabei angezeigten Meldungen können Sie im laufenden Betrieb jederzeit mit dem Kommando `dmesg` anzeigen.

Beachten Sie, dass der durch `dmesg` dargestellte Zwischenspeicher für Kernel-Meldungen begrenzt ist. Wenn dieser Speicher voll ist, werden die älteren Meldungen überschrieben. Da der Kernel auch im laufenden Betrieb Meldungen produziert, werden Sie ein vollständiges Protokoll der Startmeldungen mit `dmesg` nur unmittelbar nach dem Systemstart erhalten. Aus diesem Grund werden die Kernel-Meldungen bei den meisten Distributionen auch in eine Datei im Verzeichnis `/var/log/` geschrieben.

Sobald der Kernel läuft, beginnt der Init-V-Prozess. Auch dessen Meldungen werden in einer Datei im Verzeichnis `/var/log/` gespeichert.

Log-Dateien haben die unangenehme Eigenschaft, dass ihre Größe allmählich ins Uferlose wächst. Daher ist es hin und wieder notwendig, mit einer neuen Log-Datei zu beginnen. Die bisherige Log-Datei sollte bei dieser Gelegenheit

umbenannt und möglichst auch komprimiert werden. Das ergibt dann Dateien wie `messages.1.gz`, `messages.2.gz` etc.

Bei den meisten Distributionen ruft ein `cron`-Job dazu regelmäßig das Programm `logrotate` auf. Dieses Programm wird durch die Dateien `/etc/logrotate.conf` und `/etc/logrotate.d/*` gesteuert.

Kapitel 12

Das X Window-System

Einführung

Das X Window-System (kurz X) stellt eine Sammlung von Funktionen und Protokollen dar, mit deren Hilfe grafische Informationen auf dem Bildschirm ausgegeben und Maus und Tastatur verwaltet werden. Diese Funktionen stehen auch im Netzwerk zur Verfügung. XFree86 ist eine freie Implementierung dieses X Window-Systems. Die Treiber von XFree86 funktionieren grundsätzlich in Kombination mit beinahe jeder marktüblichen Grafikkarte. Allerdings lässt die Unterstützung neuerer Grafikkarten meist länger auf sich warten. Das liegt einerseits in der Komplexität der Treiber und andererseits daran, dass manche Firmen keine Informationen herausgeben. So bleiben manchmal Funktionen der Grafikkarten ungenutzt.
Zu XFree86 gibt es sehr ausführliche `man`-Texte: Unter anderem existieren Seiten zu `XFree86` (Überblick), zu `Xserver` (allgemeine Optionen für den Betrieb von X) und `XF86Config` (Aufbau der Konfigurationsdatei). Weitere Informationen zum XFree86-Projekt und zur gerade aktuellen Version finden Sie im Internet unter *http://www.xfree86.org*.

Mit der Ende Februar 2004 freigegebenen Version 4.4 hat sich die Lizenz von XFree86 geändert. Die neue Lizenz ist möglicherweise inkompatibel mit der GPL. Aus diesem Grund nutzen die meisten aktuellen Linux-Distributionen nun den alternativen X-Server des X.Org-Projekts (Fedora Core ab Version 2, Suse ab 9.2, Mandriva ab 10.1). Debian verwendet jedoch nach wie vor XFree86.

Dieses Kapitel ist damit zum Glück nicht obsolet, die erste Version des X.Org-Servers ist weitgehend kompatibel zu XFree86. Eine Zusammenfassung der Unterschiede finden Sie ab Seite 630.

Das X Window-System (kurz X) bezeichnet eigentlich nur die Basisfunktionen zum Zeichnen von Punkten, Linien, Rechtecken etc. X stützt sich auf ein

Netzwerkprotokoll, das es ermöglicht, ein X-Programm auf einem Rechner auszuführen (X-Client) und die Ergebnisse via Netzwerk auf einem anderen Rechner darzustellen (X-Server). XFree86 4.0 basiert auf X11R6.3. X ist die Basis für eine grafische Benutzeroberfläche für Linux. X stellt aber selbst keine Benutzeroberfläche zur Verfügung! Das Aussehen und die Bedienung von X-Programmen hängen davon ab, welche Bibliotheken bei der Programmierung eingesetzt wurden (z. B. GTK bei Gnome-Programmen, QT bei KDE-Programmen) und welcher Window Manager aktiv ist. Der X-Server stellt somit die Schnittstelle zwischen dem X Window-System und der Hardware (Grafikkarte, Tastatur, Maus) her. Seit XFree86 4.0 ist der Server modularisiert. Das bedeutet, dass der eigentliche Server durch ein Modul mit den spezifischen Funktionen für die jeweilige Grafikkarte ergänzt wird.

Der X-Server ist in der Lage, einen **virtuellen Bildschirm** zu verwalten, der größer ist als der tatsächlich am Monitor sichtbare Bildausschnitt. Die Veränderung des gerade sichtbaren Ausschnitts erfolgt verzögerungsfrei, sobald Sie die Maus an den Rand des Bildschirms bewegen. Da das Arbeiten mit einem virtuellen Bildschirm gewöhnungsbedürftig ist, wird bei der X-Konfiguration normalerweise darauf verzichtet. Ein **virtueller Desktop** ist dagegen eine vergrößerte Benutzeroberfläche, die quasi aus mehreren neben- und übereinander angeordneten Bildschirmen besteht. Sie können aber immer nur einen dieser virtuellen Bildschirme sehen. Virtuelle Desktops werden vom Window Manager oder vom Desktop-System (KDE, Gnome) verwaltet, nicht von X.

Der **Window Manager** ist ein X-Programm, das für die Verwaltung der Fenster zuständig ist. Sie können mit dem Window Manager andere Programme starten, zwischen Fenstern wechseln, Fenster verschieben und schließen etc. Es ist wichtig, sich vor Augen zu halten, dass diese Aufgaben vom Window Manager und nicht von X selbst erledigt werden. Gebräuchliche Window Manager sind `twm`, `fvwm`, `sawfish` etc. (Nicht alle Window Manager werden bei jeder Distribution mitgeliefert.)

Desktop-Systeme wie KDE oder Gnome gehen noch über die Funktion eines Window Managers hinaus. Aufgrund grafischer Bibliotheken sehen KDE- bzw. Gnome-Anwendungsprogramme einheitlich aus und sind auch einheitlich zu bedienen. Die Kommunikation zwischen den Programmen ist so weit standardisiert, dass beispielsweise Drag&Drop-Kommandos zwischen unterschiedlichen Programmen möglich sind etc. – das fühlt sich fast an wie Windows. Aufgrund der weiten Verbreitung von KDE und Gnome ist es mittlerweile unüblich geworden, X nur mit einem Window Manager (aber ohne ein zusätzliches Desktop-System) zu verwenden. Es gibt aber Fälle, wo dies auch heute noch sinnvoll ist – z. B. wenn Sie auf einem älteren Rechner mit wenig RAM oder einer langsameren CPU arbeiten. KDE und Gnome haben nämlich den Nachteil, dass sie ziemlich verschwenderisch mit den Rechnerressourcen umgehen.

12.1 Monitor-Hardware

Flachbildschirme verdrängen Röhrenmonitore zwar mehr und mehr, dennoch müssen Sie die Funktionsweise eines Röhrenmonitors im Prinzip verstanden haben, um die Konfigurationsparameter der Grafikkarte besser zu verstehen. Bei einem Röhrenmonitor kommt ein Bild zustande, indem ein Elektronenstrahl zeilenweise die ganze Bildschirmoberfläche überstreicht und dabei winzige Punkte zum Leuchten bringt. Um ein flimmerfreies Bild zu ermöglichen, wird dieser Vorgang mindestens 60-mal, besser 70- bis 100-mal pro Sekunde wiederholt.

Der Elektronenstrahl beginnt in der linken oberen Ecke des Bildschirms. Von dort bewegt er sich nach rechts, überstreicht also die oberste (erste) Zeile. Anschließend springt er zurück zum Beginn der zweiten Zeile. Während dieses horizontalen Rücklaufs bleibt das Bild dunkel, so dass er auf dem Bildschirm nicht sichtbar ist. Der horizontale Rücklauf wird durch den so genannten **Horizontal-Synchronisations-Impuls** (HSync) ausgelöst, den die Grafikkarte an den Monitor schickt.

Nachdem auf diese Weise alle Zeilen nacheinander überstrichen wurden, befindet sich der Elektronenstrahl in der rechten unteren Ecke des Bildschirms. Nun wird durch den **Vertikal-Synchron-Impuls** (VSync) der vertikale Rücklauf in die linke obere Ecke des Bildes ausgelöst. Natürlich ist der Elektronenstrahl auch während dieser Zeit dunkel geschaltet.

Die Kenndaten des Bildaufbaus werden durch drei Frequenzen charakterisiert (Einheit Hertz = Schwingungen pro Sekunde, abgekürzt Hz):

- Die **vertikale Bildfrequenz** gibt an, wie oft pro Sekunde die ganze Bildschirmfläche neu gezeichnet wird. Je größer diese Zahl ist, desto stabiler wirkt das Bild.
- Die **horizontale Zeilenfrequenz** gibt an, wie viele Zeilen pro Sekunde vom Elektronenstrahl überstrichen werden. Bei einer Bildauflösung von 640 $*$ 480 Punkten (480 Zeilen à 640 Punkte) und einer Bildfrequenz von 60 Hz beträgt die horizontale Zeilenfrequenz 31,5 kHz. (Nachrechnen ergibt: $60\ Hz * 480 = 28,8\ kHz$. Die etwas höhere Frequenz ist notwendig, weil auch etwas Zeit für den vertikalen Rücklauf berücksichtigt werden muss.) Eine zu hohe Zeilenfrequenz tut dem Monitor übrigens gar nicht gut. Moderne Monitore sollten das erkennen und den Elektronenstrahl automatisch abschalten; bei älteren Modellen konnte man durch falsche HSync-Werte schon mal den Monitor schrotten.
- Die **Pixelfrequenz (Video-Bandbreite)** gibt die Anzahl der Bildpunkte (Pixel) an, die während einer Sekunde von der Grafikkarte an den Monitor übertragen werden. Bei 640x480 Punkten und 60 Hz ergibt sich ein Wert von 18,4 MHz. Wegen des durch den vertikalen und horizontalen Rücklauf bedingten Overheads beträgt die Pixelfrequenz aber tatsächlich ca. 25 MHz.

Bei Flachbildschirmen (LCD, Liquid Crystal Display) erfolgt der Bildaufbau, indem die Steuerungselektronik kleine Teile der Bildoberfläche durchsichtig bzw. undurchsichtig macht. Ein zeilenweiser Bildaufbau ist an sich nicht mehr erforderlich.

LCD-Monitore mit analogem Signaleingang verarbeiten aus Kompatibilitätsgründen dasselbe Eingangssignal wie Röhrenmonitore. Allerdings ist die Umwandlung des analogen Videosignals in die digitale Pixeldarstellung verlustbehaftet. Eine bessere Bildqualität wird erreicht, wenn das Signal digital übertragen wird. Das erfordert DVI-Anschlüsse (*Digital Visual Interface*) bei Grafikkarte und Monitor. Außerdem haben Flachbildschirme nur eine optimale Auflösung, die konstruktionsbedingt der Anzahl der Pixel in horizontaler und vertiklaer Richtung entspricht. Alle anderen Auflösungen werden interpoliert, was zu einer leichten Bildunschärfe führt. Im Gegensatz zu Röhrenmonitoren ist auch die Bildfrequenz meist fest vorgegeben(60 – 75 Hz). Bei falscher Bildfreqenz schaltet das LCD ab.

Die meisten modernen Monitore übermitteln über das Monitorkabel ihre Eckdaten (z. B. die maximale Zeilenfrequenz) an die Grafikkarte (DCC (Display Data Channel). Das ermöglicht die automatische Erkennung des Monitors und hilft so bei der richtigen Konfiguration der Grafikkarte.

12.2 X starten und beenden

Üblicherweise ist Linux so konfiguriert, dass X beim Systemstart automatisch ausgeführt wird und sich mit einer Login-Box meldet. Für diesen automatischen X-Start ist der Init-V-Runlevel 5 verantwortlich (siehe auch Seite 604). Wenn X nicht schon läuft, bestehen zwei Möglichkeiten für einen manuellen Start. Bis die Konfiguration abgeschlossen ist, sollten Sie sich für die erste Variante entscheiden.

Mit dem Kommando `startx` starten Sie X direkt. Es erscheint keine Login-Box. Der Benutzer, der `startx` ausführt, ist auch der Benutzer unter X. Einige Distributionen (z. B. Mandriva, Red Hat) verlassen sich darauf, dass beim Start von X bereits ein Font-Server läuft. Wenn Sie X manuell starten, ist das nicht der Fall. Deswegen müssen Sie ggf. auch den Font-Server manuell mittels `/etc/init.d/xfs start` starten. (Das darf nur `root`.) Der Display-Manager ist für die grafische Login-Box verantwortlich. Derzeit kommen dabei je nach Distribution drei unterschiedliche Programme zum Einsatz: `xdm`, die KDE-Variante `kdm` oder die Gnome-Variante `gdm`. Diese Programme werden allerdings nicht direkt gestartet, sondern durch einen Runlevel-Wechsel. Das können Sie als `root` in einer Textkonsole durch das Kommando `init 5` erreichen.

Wie X richtig beendet wird, hängt ein wenig davon ab, wie es gestartet wurde. Generell sollten Sie versuchen, KDE, Gnome oder den aktiven Window Manager über das dafür vorgesehene Menükommando zu beenden. Falls Sie X zuvor

manuell gestartet haben, finden Sie sich jetzt in der Textkonsole wieder, d. h. X wurde tatsächlich beendet.

Wurde X dagegen über einen Display-Manager gestartet, erscheint nun wiederum die grafische Login-Box. X läuft aber weiter! Wie Sie X nun tatsächlich beenden können, hängt vom Display-Manager ab. Manche Display-Manager bieten die Möglichkeit, vom aktuellen Runlevel 5 zum Runlevel 3 zu wechseln und damit sowohl den Display-Manager als auch X endgültig zu beenden. Beim KDE-Display-Manager klicken Sie dazu den Button *Beenden* an und wählen die Option *Textkonsole* aus.

Wenn diese Möglichkeit nicht gegeben ist, müssen Sie das Kommando `init 3` selbst ausführen. Sie können dieses Kommando unter X (nur als `root`) in einem Shell-Fenster ausführen. Die andere Variante besteht darin, dass Sie mit Strg+Alt+F1 in eine Textkonsole wechseln, sich dort als `root` einloggen und jetzt `init 3` ausführen.

Tipp

Sie können X auch mit Strg+Alt+Backspace beenden. Das ist allerdings nicht die feine Art, weil Sie damit KDE oder Gnome keine Chance geben, Aufräumarbeiten durchzuführen, den aktuellen Desktop-Zustand zu speichern etc. Die Tastenkombination ist aber praktisch, wenn X aus irgendeinem Grund nicht bedienbar ist (z. B. weil der Monitor nicht synchronisiert ist und kein stabiles Bild liefert). Falls X über einen Display-Manager gestartet wurde, führt diese Tastenkombination zurück zur Login-Box. (X wird dazu automatisch neu gestartet.) In diesem Fall wechseln Sie mit Strg+Alt+F1 in die erste Textkonsole und können dort `init 3` ausführen.

Wenn Sie X manuell starten (also mit `startx`), werden in der Textkonsole zahlreiche Meldungen, Warnungen und eventuell auch Fehlermeldungen angezeigt. Parallel dazu wird bei jedem X-Start ein Startprotokoll in der Datei `/var/log/XFree86.*.log` erstellt. Dieses Protokoll enthält ausführliche Informationen darüber, welche Konfigurationsdatei verwendet wurde, welche Module geladen wurden, welche Probleme dabei eventuell aufgetreten sind, welche Grafikmodi aus welchen Gründen verworfen wurden etc. Wenn es bei der X-Konfiguration Probleme gibt, sollte dieses Protokoll bei der Fehlersuche helfen. Einträge innerhalb der Logging-Datei sind durch folgende Codes gekennzeichnet:

`(**)`: Einstellung aus der Konfigurationsdatei
`(++)`: Einstellung aus der Kommandozeile
`(==)`: XFree86-Default-Einstellung
`(--)`: Einstellung, die sich aus erkannter Hardware ergibt
`(!!)`: Hinweis
`(II)`: Hinweis
`(WW)`: Warnung
`(EE)`: Fehler

12.3 Konfiguration des X-Servers

Die Konfiguration des X-Servers erfolgt normalerweise während der Installation. Wenn X auf Ihrem Rechner bereits zufrieden stellend läuft, können Sie diesen Abschnitt getrost überspringen. Es wird davon ausgegangen, dass X noch nicht läuft.

Die Konfiguration von XFree86 erfolgt üblicherweise durch die Datei `/etc/X11/XF86Config`. Es gibt aber eine Menge weiterer möglicher Orte für diese Datei. (Eine vollständige Liste aller zulässigen Verzeichnisse erhalten Sie mit `man XF86Config`.) Zulässig sind unter anderem die Verzeichnisse `/etc, /etc/X11/XF86Config* und /usr/X11R6/etc/X11/XF86Config*`. X berücksichtigt ausschließlich die erste Konfigurationsdatei, die es findet! Normalerweise sollten Sie die Datei `XF86Config` nicht manuell verändern, sondern stattdessen das Konfigurationswerkzeug Ihrer Distribution einsetzen (`redhat-config-xfree86`, `sax2` oder `XFree86 -configure` bzw. `xf86cfg` bzw. `xf86config`).

Die Datei `XF86Config` ist in mehrere Abschnitte gegliedert, die mit `Section "name"` eingeleitet und mit `EndSection` abgeschlossen werden. Dabei sind die folgenden Abschnittsnamen vorgesehen:

XF86Config-Abschnitte

`Monitor`	Monitordaten
`Device`	Konfiguration der Grafikkarte
`Screen`	Bildschirmauflösung
`Modes`	Video-Modi (optional)
`Files`	Dateinamen (z. B. Font-Verzeichnisse)
`InputDevice`	Konfiguration von Tastatur und Maus
...	

In den meisten Abschnitten gibt es eine `Identifier`-Zeile. Diese Zeile gibt dem Abschnitt einen Namen. Die Namen werden in den Abschnitten `Screen` und `ServerLayout` verwendet, um die Einzelinformationen zusammenzuführen. Nicht so wichtig sind dagegen die diversen `Board-`, `Vendor-` und `ModelName`-Zeichenketten. Sie dienen nur zur besseren Orientierung innerhalb der Konfigurationsdatei. Von den weiteren Schlüsselwörtern werden die wichtigsten im Verlauf der folgenden Abschnitte beschrieben. Eine vollständige Referenz enthält die `man`-Seite zu `XF86Config`.

12.3.1 Monitor-Abschnitt

Im `Monitor`-Abschnitt sind im Regelfall nur zwei Angaben erforderlich: der zulässige Bereich für die horizontale Zeilenfrequenz (in kHz) und der zulässige Bereich für die Bildfrequenz (in Hz). Die folgenden Angaben gelten für einen typischen 19-Zoll-Röhrenmonitor:

```
HorizSync     27-102   # 27 bis 102 kHz (Zeilen/sec)
VertRefresh   50-160   # 50 bis 160 Hz  (Bilder/sec)
```

Flachbildschirme (LCD-Monitore) zeichnen sich dadurch aus, dass der Frequenzbereich für die Bildfrequenz (*VertRefresh*) viel kleiner ist. Die folgenden Daten gelten für einen 20-Zoll-LCD-Monitor mit einer Auflösung von $1600 * 1200$ Pixeln und einer maximalen Bildfrequenz von 75 Hz:

```
HorizSync    30-95     # 30 bis 95 kHz (Zeilen/sec)
VertRefresh  58-78     # 58 bis 78 Hz  (Bilder/sec)
```

Optional können Sie zwischen den Schlüsselwörtern `Mode` und `EndMode` zusätzliche Grafikmodi angeben, die speziell für diesen Monitor gelten. Alternativ können Sie diese Modi auch mit `ModeLine` in kompakter Form angeben oder mit `UseModes` auf einen separaten `Modes`-Abschnitt verweisen.

12.3.2 Device-Abschnitt (Grafikkarte)

Das wichtigste Schlüsselwort im Device-Abschnitt ist `Driver`. Es bestimmt, welches Modul geladen werden soll. Die zur Auswahl stehenden Grafiktreiber befinden sich im Verzeichnis `/usr/X11R6/lib/modules/drivers`. Falls mehrere PCI-Grafikkarten in den Rechner eingebaut sind, können Sie mit `BusID` genau angeben, welche Sie meinen. Die drei Ziffern geben den PCI-Bus, die Device-Nummer und die Funktion an. Die korrekten Werte können Sie herausfinden, indem Sie `X -scanpci` ausführen. (X darf zu diesem Zeitpunkt nicht laufen.) Wenn Sie nicht wissen, welche Grafikkarte Sie haben, probieren Sie es als `root` mit dem Kommando `lspci`. Leider geht aus dem Ergebnis nicht immer auch der erforderliche Grafiktreiber hervor. Für neuere ATI-Karten muss beispielsweise nicht der *ati*-, sondern der *radeon*-Treiber verwendet werden. Hilfreich bei der Treiberauswahl sind das Kapitel *Drivers* der XFree86-Release-Notes sowie das Dokument *Driver Status*:

http://www.xfree86.org/current/RELNOTES.html
http://www.xfree86.org/current/Status.html

Wenn Sie Pech haben, wird Ihre neue Grafikkarte von XFree86 noch gar nicht oder nur teilweise unterstützt. Manchmal scheitert es nur an der richtigen Erkennung der Grafikkarte: Sie haben also im *Device*-Abschnitt das richtige Modul angegeben, aber XFree86 erkennt die Grafikkarte nicht. In diesem Fall können Sie versuchen, im *Device*-Abschnitt mit *ChipId* die ID-Nummer einer kompatiblen Karte einzusetzen (z. B. `ChipId "0x1234"`).

12.3.3 Screen-Abschnitt (Auflösung, Farbanzahl)

Der `Screen`-Abschnitt verbindet den Monitor und die Grafikkarte und gibt an, in welcher Auflösung und mit wie vielen Farben die Grafikkarte verwendet werden soll. In jeder `XF86Config`-Datei muss es einen `Screen`-Abschnitt geben.

Die Schlüsselwörter `Device` und `Monitor` verweisen auf die oben schon definierte Grafikkarte und den Monitor. `DefaultDepth` gibt an, wie viele Farben per Default zur Verfügung stehen. Die Angabe erfolgt in Bit pro Pixel. Innerhalb des `Screen`-Abschnitts können mehrere `Display`-Unterabschnitte angegeben werden, je einer für jede Farbkonfiguration (Schlüsselwort `Depth`). Im Beispiel unten sind zwei Modi definiert, einer mit 16 Bit und einer mit 24 Bit pro Pixel. Welcher dieser Modi aktiv ist, bestimmt die `DefaultDepth`-Option.

```
Device        "Device1"
Monitor       "Monitor1"
DefaultDepth  24
SubSection "Display"
  Depth       16
  Modes       "1280x1024"
EndSubSection
SubSection "Display"
  Depth       24
  Modes       "1280x1024" "1024x768" "800x600" "640x480"
EndSubSection
```

In der optionalen `Modes`-Zeile kann die gewünschte Auflösung angegeben werden. Wenn die Zeile weggelassen wird oder eine ungültige Auflösung enthält, entscheidet sich X automatisch für die bestmögliche Auflösung, die für den Monitor und die Grafikkarte geeignet ist. XFree86 verfügt dazu über eine *Mode*-Datenbank. Diese enthält die Timing-Parameter für die von der VESA definierten Auflösungen. Es ist leider nicht möglich, die gewünschte Bildfrequenz direkt einzustellen. X entscheidet darüber selbst und berücksichtigt dabei die Eckdaten des Monitors und der Grafikkarte.

Wenn in der `Modes`-Zeile mehrere gültige Auflösungen angegeben werden, verwendet X die erste als Default-Einstellung. Zwischen den Auflösungen können Sie nun im laufenden Betrieb mit (Strg)+(Alt)+(+) („+“ auf dem numerischen Tastenblock) wechseln. In den Modi mit geringerer Auflösung gilt die höchste zulässige Auflösung als virtueller Bildschirm. Optional kann in jedem `Display`-Abschnitt die Größe des virtuellen Bildschirms eingestellt werden. `Virtual 1600 1200` bewirkt beispielsweise, dass ein virtueller Bildschirm von $1600 * 1200$ Punkten verwaltet wird, unabhängig davon, mit welcher Auflösung der Monitor tatsächlich verwendet wird.

12.3.4 Grafikmodus selbst definieren

X verfügt über eine Datenbank mit allen von der VESA definierten Bildschirmauflösungen und Bildfrequenzen. Solange Sie einen dieser Grafikmodi verwenden (etwa durch die Zeile `Modes "1280x1024"`), brauchen Sie sich nicht darum zu kümmern, wie derartige Grafikmodi intern definiert sind. Wenn Sie allerdings eine andere Auflösung verwenden oder eine bestimmte Bildfrequenz erreichen

möchten, müssen Sie die Timing-Parameter des gewünschten Grafikmodus selbst angeben. Dazu gibt es mehrere Möglichkeiten:

- Sie können einen bestimmten Grafikmodus mit `ModeLine` im `Monitor`-Abschnitt angeben.
- Sie können den oder die Grafikmodi aber auch in einem eigenen `Modes`-Abschnitt angeben und darauf mit `UseMode` verweisen.

Egal, wo die Modi definiert sind – jeder Grafikmodus bekommt einen Namen. Üblicherweise beschreibt der Name die Auflösung (also `1024x768`). Welcher der definierten Modi tatsächlich verwendet werden soll, bestimmt die `Modes`-Zeile im `Screen`-Abschnitt.

Ein Grafikmodus wird durch seinen Namen und neun Zahlenwerte bestimmt. Die folgende Zeile zeigt ein Beispiel:

```
ModeLine "640x480"  25.175   640 664 760 800   480 491 493 525
```

Damit wird ein Grafikmodus mit $640 * 480$ Pixeln beschrieben. Die Zeichenkette `"640x480"` ist gleichzeitig auch der Name dieses Modus. Der Zahlenwert 25.175 gibt die Pixelfrequenz (Videobandbreite) in MHz an. Die nächsten vier Werte (in der Einheit Pixel) betreffen das horizontale Timing: Eine einzelne Bildschirmzeile mit 640 *sichtbaren* Pixeln wird in Wirklichkeit aus 800 *virtuellen* Pixeln zusammengesetzt. Die ersten 640 Pixel werden tatsächlich angezeigt. Während der verbleibenden 160 Pixel wird der Elektronenstrahl durch den HSync-Impuls zurück an den Beginn der nächsten Zeile bewegt. Während dieser Zeit hat der Elektronenstrahl die Intensität 0. Die vier Werte kommen also wie folgt zustande:

640	640 Bildschirmpixel anzeigen
664	24 weitere Pixel dunkel tasten
760	96 Pixel lang einen HSync-Impuls erzeugen
800	nochmals 40 Pixel dunkel tasten, d. h. insgesamt 800 virtuelle Punkte

Analog wie beim horizontalen Timing sind auch die Angaben für das vertikale Timing (Einheit Bildschirmzeilen) zu interpretieren:

480	480 Zeilen anzeigen
491	11 Zeilen dunkel tasten
493	2 Zeilen lang einen VSync-Impuls erzeugen
525	nochmals 32 Zeilen dunkel tasten, d. h. insgesamt 525 virtuelle Zeilen

Aus den jeweils letzten Werten der Vierergruppen und der Pixelfrequenz ergeben sich übrigens die horizontale Zeilenfrequenz und die vertikale Bildfrequenz: 25.175 MHz dividiert durch 800 Pixel pro Zeile ergibt eine Zeilenfrequenz von

31.469 kHz. Die Zeilenfrequenz, dividiert durch 525 Zeilen pro Bild, liefert die vertikale Bildfrequenz von 60 Hz.

Grundsätzlich sollte es anhand dieser Informationen möglich sein, eigene `ModeLine`-Definitionen durchzuführen. Einzig bei den Timing-Angaben für HSync und VSync muss meist ein wenig experimentiert werden.

Wenn Sie nur die Bildfrequenz erhöhen möchten, gehen Sie am besten von einem existierenden Modus aus (hier für $1152 * 864$ Pixel bei 75 Hz mit einer Videobandbreite von $1474 * 895 * 75 = 98.942.250$ Hz = 98,9 MHz):

```
ModeLine "mymode" 98.9 1152 1208 1368 1474 864 865 875 895
```

Nun multiplizieren Sie die jeweils letzten Werte der beiden Vierergruppen und die gewünschte Bildfrequenz. Wenn Sie also $1152 * 864$ bei 85 Hz haben wollen, multiplizieren Sie $1474 * 895 * 85 = 112.134.550$ Hz = 112,1 MHz. Das Ergebnis ist die Videobandbreite, also der erste Wert der *Modes*-Zeile. Die neue `ModeLine` sieht damit so aus:

```
ModeLine "mymode" 112.1 1152 1208 1368 1474 864 865 875 895
```

Wenn Sie sich intensiver mit der Komposition eigener `ModeLines` auseinander setzen möchten, empfiehlt sich die Lektüre des XFree86 Video Timings HOWTO unter *http://tldp.org/HOWTO/XFree86-Video-Timings-HOWTO/*.

Meist ist es aber einfacher, sich im Internet eine geeignete `ModeLine`-Zeile zu suchen. Besonders für die LCDs diverser Notebooks ist die Webseite *http://www.linux-laptop.net/* ein guter Startpunkt für die Suche.

Sie können aber auch mit einer Modeline beginnen, die zumindest halbwegs passt und diese dann etwas „tunen". Dazu verwenden Sie einen Modeline-Generator wie z. B. `xvidtune`. Stellen Sie vorher absolut sicher, dass in der Monitor-Sektion der aktuell gültigen XF86Config unter HorizSync bzw. VertRefresh die exakten Werte des angeschlossenen Monitors stehen (aus dem Handbuch). `xvidtune` liegt ohnehin jeder Distribution bei. Es begrüßt Sie zunächst mit einer Warnung, die Sie nach Kenntnisnahme mit OK bestätigen können. Mit *Next* bzw. *Prev* kann der Auflösungsmodus gewechselt werden. Wechseln Sie ggf. in den Modus, dessen Modelines Sie optimieren möchten. Rechts unten sehen Sie den jeweils aktuellen Refresh-Wert Vertical Sync (Hz). Mit *Wider/Narrower* bzw. *Shorter/Taller* können Sie diesen Wert im Folgenden beeinflussen. Die veränderten Einstellungen werden durch einen Klick auf *Test* aus Sicherheitsgründen nur kurzzeitig aktiv. Beim Klick auf *Apply* schaltet `xvidtune` dagegen dauerhaft auf die neuen Modeline-Werte.

Machen Sie am Anfang nur kleine Schritte und testen Sie diese immer wieder kurz. Sollte das Bild über die Bildschirmränder hinausragen oder andere Darstellungsfehler aufweisen, können Sie das Bild nachjustieren. Sollte die Bildlage bei der optimalen Refresh-Rate noch verbesserungsbedürftig sein, können Sie diese auch direkt am Monitor verändern. *Show* gibt die (auflösungsabhängige)

optimale Modeline schließlich auf dem X-Term aus, auf dem Sie `xvidtune` zuvor gestartet haben.

Setzen Sie diese Zahlenreihe an der entsprechenden Stelle Ihrer XF86Config ein. Je nachdem, wie viele Modelines Ihr Konfigurationsprogramm zuvor in der Monitor-Sektion erzeugt hat, ist es durchaus nicht ganz trivial, die zu ersetzende Modeline zu finden. Richten Sie sich dabei am besten nach dem ersten Wert (Pixel Clock in MHz), der in der mit `xvidtune` erzeugten Modeline nach der Auflösungsangabe folgt. Wenn Sie in der Monitor-Sektion eine Modeline finden, die diesen Wert enthält und darüber hinaus auch in der Auflösung übereinstimmt, dann dürfte es sich in der Regel um die zu ersetzende Modeline handeln.

12.3.5 Files-Abschnitt

Im `Files`-Abschnitt werden die Orte diverser Verzeichnisse angegeben, aus denen der X-Server Dateien lädt. Im Regelfall enthält dieser Abschnitt eine `RgbPath`-Zeile, die auf eine Datei mit den Namen verschiedener Farben verweist, eine `ModulePath`-Zeile, die auf ein Verzeichnis mit X-Server-Modulen verweist, und schließlich zahllose `FontPath`-Zeilen mit den Orten diverser Font-Verzeichnisse.

```
Section "Files"
  RgbPath       "/usr/X11R6/lib/X11/rgb"
  ModulePath    "/usr/X11R6/lib/modules"
  FontPath      "/usr/X11R6/lib/X11/fonts/misc:unscaled"
  FontPath      "/usr/X11R6/lib/X11/fonts/local"
  ...
EndSection
```

12.3.6 Modules-Abschnitt

Im `Modules`-Abschnitt von `XF86Config` geben Sie mit dem Schlüsselwort `Load` an, welche Module der X-Server verwenden soll:

```
Section "Module"
  Load        "extmod"
  ...
EndSection
```

Die Module befinden sich in den Dateien `extensions` und `fonts` im Verzeichnis `/usr/X11R6/lib/modules/`. Das `extmod`-Modul sollte immer verwendet werden, weil andernfalls einige X-Server-Erweiterungen nicht zur Verfügung stehen. Diese Zusatzfunktionen werden von zahlreichen Programmen vorausgesetzt. Font-Module werden nur benötigt, wenn kein Font-Server verwendet wird.

12.3.7 ServerFlags-Abschnitt

Im `ServerFlags`-Abschnitt können Optionen angegeben werden, die das Verhalten des X-Servers beeinflussen:

```
Section "ServerFlags"
  Option        "DontZap" "on"
EndSection
```

Im Folgenden werden nur die drei wichtigsten Optionen beschrieben. Eine vollständige Beschreibung aller Optionen finden Sie mit `man XF86Config`.

`DontZap` (Default `off`): Die Einstellung `on` deaktiviert die Tastenkombination (Strg) + (Alt) + (Backspace) zum sofortigen Beenden des X-Servers.

`DontVTSwitch` (Default `off`): Die Einstellung `on` deaktiviert die Tastenkombination (Strg) + (Alt) + (Fn) zum Wechsel in eine Textkonsole.

`AllowMouseOpenFail` (Default `off`): Die Einstellung `on` bewirkt, dass X selbst dann gestartet wird, wenn die Initialisierung oder Erkennung der Maus scheitert.

12.3.8 Umgang mit XFree86-inkompatiblen Grafikkarten

Die Zusammenarbeit zwischen XFree86 und den Herstellern von Grafikkarten hat lange Zeit gut funktioniert. In den letzten Jahren hat sich die Situation leider wieder verschlechtert. Aus diesem Grund fehlen in XFree86 vollständige Treiber für die meisten aktuellen Chipsätze von ATI, Matrox, NVIDIA etc. Das bedeutet nicht, dass Sie derartige Grafikkarten nicht nutzen können. Es gibt verschiedene Möglichkeiten, die Grafikkarte dennoch zu verwenden:

- Soweit vorhanden, können Sie auf einen unvollständigen oder kompatiblen XFree86-Treiber für Ihre Karte zurückgreifen. Beispielsweise können fast alle NVIDIA-Grafikkarten mit dem XFree86-Modul `nv` betrieben werden, sehr viele ATI-Radeon-Karten mit dem Modul `radeon` etc. Allerdings ist die XFree86-Unterstützung je nach Modell der Grafikkarte ziemlich limitiert.
- Wenn es keinen spezifischen Treiber für Ihre Karte gibt, können Sie auf einen der folgenden XFree86-Treiber zurückgreifen: `vesa`, `fbdev` oder `vga`. Damit lassen sich beinahe alle Grafikkarten betreiben. Alle drei Treiber haben aber den Nachteil, dass Sie weder 2D- noch 3D-Funktionen zur Beschleunigung der Ausgabe nutzen.
- Schließlich gibt es für immer mehr Grafikkarten herstellerspezifische Binärtreiber. Diese sind zwar kostenlos, stehen aber leider nur in Binärform zur Verfügung (und zumeist nur für Rechner mit Intel-Pentium-kompatiblen Prozessoren). In der Regel verbieten die Lizenzbedingungen die Integration bzw. Auslieferung der Treiber mit einer Linux-Distribution. Daher ist eine manuelle Installation erforderlich.

Sobald die meist unkomplizierte Installation gelingt, kann die Grafikkarte optimal genutzt werden – allerdings nur bis zum nächsten Kernel-Update! Die Binärtreiber sind nämlich über ein spezielles Modul mit dem Kernel verbunden. Dieses Kernel-Modul muss bei jedem Kernel-Update neu erstellt (kompiliert) werden.

Die Verwendung eines Flachbildschirms mit einem DVI-Kabel ist ein Problem für sich. Manchmal kann XFree zwar die Grafikkarte richtig steuern, es gelingt aber nicht, ein digitales Ausgangssignal zu erzeugen. Bei `nv`-Modulen für NVIDIA-Grafikkarten existiert eine Option, um den DVI-Ausgang der Grafikkarte explizit zu aktivieren:

```
Section "Device"
  ...
  Option "FlatPanel" "on"
EndSection
```

Ansonsten kann es während der Konfigurationsarbeiten zweckmäßig sein, vorübergehend den analogen Signaleingang einzusetzen. Beachten Sie aber, dass bei manchen Grafikkarten für einen Wechsel zwischen Analog- und Digitalmodus ein Neustart erforderlich ist. Die Karte erkennt beim Rechnerstart, ob ein analoges oder ein DVI-Kabel angeschlossen ist.

12.3.9 VESA- und Framebuffer-Treiber

VESA-Treiber: Mit dem `vesa`-Treiber können Sie alle VESA-Modi Ihrer Grafikkarte nutzen. Kurz einige Hintergrundinformationen: Die *Video Electronics Standard Association* hat eine Reihe von Grafikmodi für Standardauflösungen normiert. Jeder Modus ist durch die folgenden Eckdaten bestimmt: Auflösung (z. B. $1280 * 1024$ Pixel), Farbtiefe und Bildfrequenz. Fast alle Grafikkarten unterstützen neben eigenen Grafikmodi auch eine Menge VESA-Modi.

Wie die nächsten Zeilen zeigen, ist die Verwendung des *VESA*-Treibers denkbar einfach. Sofern die restliche Konfigurationsdatei korrekt ist, werden alle VESA-Modi berücksichtigt, die die Grafikkarte unterstützt und die der Monitor darstellen kann.

```
# in /etc/X11/XF86Config
...
Section "Device"
  Identifier    "myDevice"
  Driver        "vesa"
EndSection
```

Framebuffer-Treiber: Der `fbdev`-Treiber greift direkt auf den Speicher (Framebuffer) der Grafikkarte zu. Der Treiber setzt damit noch eine Ebene tiefer an

als der VESA-Treiber. Er sollte mit fast allen Grafikkarten funktionieren, sofern der Linux-Kernel mit Framebuffer-Unterstützung kompiliert wurde. Dass diese Unterstützung vorhanden ist, erkennen Sie daran, dass die Datei `/proc/fb` existiert.

Eine grundlegende Voraussetzung für die Nutzung des Treibers besteht allerdings darin, dass bereits beim Booten des Rechners der richtige VGA-Modus ausgewählt wird. Bis zum Neustart des Rechners kann X nur in dem so festgelegten Grafikmodus betrieben werden.

Zur Auswahl des Modus fügen Sie in die GRUB- oder LILO-Konfigurationsdatei die Kernel-Option `vga=`*n* ein. Die richtigen Werte (dezimal) für *n* finden Sie in der folgenden Tabelle:

	640x480	800x600	1024x768	1280x1024	1600x1200
8 bpp	769	771	773	775	796
16 bpp	785	788	791	794	798
24 bpp	786	789	792	795	799

In `XF86Config` müssen Sie lediglich die richtige *Driver*-Zeichenkette angeben:

```
# in /etc/X11/XF86Config
...
Section "Device"
  Identifier    "myDevice"
  Driver        "fbdev"
EndSection
```

12.3.10 Tastatur

Die Tastatur wird in `XF86Config` durch einen `InputDevice`-Abschnitt konfiguriert. Die folgenden Zeilen zeigen eine Minimalkonfiguration für ein deutsches Tastaturlayout (Tastatur mit 104 Tasten):

```
Section "InputDevice"
  Identifier  "myKeyboard"
  Driver      "Keyboard"
  Option      "XkbRules"    "xfree86"
  Option      "XkbModel"    "pc104"
  Option      "XkbLayout"   "de"
  Option      "XkbVariant"  "nodeadkeys"
EndSection
```

Hier einige Tipps zu den wichtigsten Optionen:

- `XkbModel` beschreibt die Tastatur. Zulässige Einstellungen sind unter anderem:

`pc101`: US-Tastatur ohne Windows-Tasten (Default)
`pc102`: Internationale Tastatur ohne Windows-Tasten
`pc104`: US-Tastatur mit Windows-Tasten
`pc105`: Internationale Tastatur mit Windows-Tasten

- `XkbLayout` beschreibt die Anordnung der Tasten auf der Tastatur. Diese ist länderabhängig:

  `us`: US-Englisch (Default)
  `de`: Deutsch
  `fr`: Französisch
  `it`: Italienisch

- `XkbVariant` ermöglicht Zusatzeinstellungen zum Tastaturlayout. Die einzige uns bekannte Einstellung lautet `nodeadkeys`. Sie bewirkt, dass die Zeichen `~ ^ ' `` unmittelbar eingegeben werden dürfen. (Per Default gelten diese Zeichen als Kompositionszeichen, d. h. ~, a ergibt das Zeichen ã.)

Wenn Sie unter KDE arbeiten, können Sie das Tastaturlayout auch im KDE-Kontrollzentrum einstellen (*Regionaleinstellungen* → *Tastaturlayout*). Unter Gnome haben Sie vergleichbare Möglichkeiten mit dem Applet *Utility* → *Tastaturbelegungsumschalter*. Wenn also Veränderungen an der Tastaturkonfiguration in `XF86Config` wirkungslos bleiben, liegt dies höchstwahrscheinlich am Desktop-System.

Je nach Distribution können unterschiedliche Tasten zur Zusammensetzung von Sonderzeichen verwendet werden. Sie können die möglichen Compose-Tasten mit `xmodmap` feststellen, indem Sie nach Tastenkombinationen für Multi-Key suchen. (Beim Ergebnis gilt die erste Spalte für die Taste ohne Shift, die zweite Spalte für die Taste mit Shift.)

```
user$  xmodmap -pk — grep -i multi˙key
KeyCode     Keysym (Keysym) ...        (Shift)
    109     0xffe4 (Control_R)         0xff20 (Multi_key)
    113     0xfe03 (ISO_Level3_Shift)  0xff20 (Multi_key)
    116     0xffec (Super_R)           0xff20 (Multi_key)
```

Das bedeutet, dass die Tastenkombinationen Shift + AltGr, Shift + Rechte Strg-Taste oder Shift + Rechte Windows-Taste als Compose-Tasten verwendet werden können. Welche Sonderzeichen wie zusammengefügt werden können, hängt vom aktiven Zeichensatz ab. Die Definitionen sind in `/usr/X11R6/lib/X11/locale` gespeichert. Die folgenden Zeilen zeigen einige Kombinationen für ISO-Latin-1 (Kommando `less /usr/X11R6/lib/X11/locale/iso8859-1/Compose`):

```
...
<Multi_key> <c> <o>                : "\251"    copyright
```

```
<Multi_key> <less> <less>          : "\253"    guillemotleft
<Multi_key> <greater> <greater>    : "\273"    guillemotright
<Multi_key> <plus> <minus>         : "\261"    plusminus
<Multi_key> <minus> <plus>         : "\261"    plusminus
<Multi_key> <slash> <u>            : "\265"    mu
<Multi_key> <u> <slash>            : "\265"    mu
```

12.3.11 xmodmap, xev, setxkbmap

Wenn Sie Probleme mit der Tastatur haben, sollten Sie zuerst `XF86Config` kontrollieren und dann (falls Sie mit KDE arbeiten) einen Blick in den Tastaturdialog des KDE-Kontrollzentrums werfen. Lässt sich die Ursache des Problems dort nicht finden, helfen vielleicht die beiden hier beschriebenen X-Programme weiter.

Das Programm `xev` dient dazu, Informationen über X-Ereignisse anzuzeigen (und jede Tastatureingabe zählt als Ereignis). Das Programm muss aus einem Shell-Fenster heraus gestartet werden. Solange das `xev`-Fenster aktiv ist, werden im Shell-Fenster Informationen über gedrückte Tasten, Mausbewegungen etc. angezeigt. Auf diese Weise können Sie feststellen, wie X intern reagiert, wenn Sie beispielsweise F1 drücken.

Das Kommando `xmodmap` gibt Auskunft darüber, welche Zeichen bzw. Funktionen einer bestimmten Taste zugeordnet sind. `xmodmap -pk` liefert die zurzeit gültige Tastaturtabelle. Dabei wird in der ersten Spalte der interne Keycode angegeben (die Nummer der Taste). In den weiteren Spalten wird die Bedeutung der Taste zuerst ohne Shift, dann mit Shift, dann mit AltGr und schließlich mit Shift+AltGr angegeben. Ein Beispiel:

```
user$ xmodmap -pk
...
38      0x0061 (a)      0x0041 (A)      0x00e6 (ae)     0x00c6 (AE)
```

Mit der Taste Nummer 38 wird der Buchstabe a bzw. A eingegeben. Wird gleichzeitig AltGr gedrückt, können Æ bzw. æ eingegeben werden.

Die Zuordnung zwischen Taste und Nummer ist von der Tastatur abhängig. Im Folgenden sehen Sie eine Liste mit den dezimalen Codes der Modifier-Tasten, die in den `xmodmap`-Listen am schwersten zu interpretieren sind. Die Nummern gelten für eine internationale Standard-PC-Tastatur mit Windows-Tasten. (Wenn Sie die Nummern anderer Tasten herausfinden möchten, verwenden Sie einfach `xev`.)

Keycodes der Modifier-Tasten (Internationale PC-Tastatur)

Shift links	50	Shift rechts	63
Strg links	37	Strg rechts	109
Windows links	115	Windows rechts	116
Alt links	64	AltGr rechts	113
CapsLock	66	Windows-Kontext-Menü rechts	117

Mit `xmodmap -pke` wird die zurzeit gültige Tastaturtabelle in der für `xmodmap` erforderlichen Syntax ausgegeben. `xmodmap datei` verarbeitet die in der Datei angegebenen Definitionen. `xmodmap -e definition` ermöglicht die Neudefinition einer einzelnen Taste:

`user$` **xmodmap -e "keycode 22 = BackSpace"**

Angesichts der guten Default-Konfiguration wird die Verwendung von `xmodmap` eigentlich nur in Ausnahmefällen erforderlich sein. Dennoch sehen viele Distributionen `Xmodmap`-Dateien vor, die beim X-Start automatisch geladen und ausgeführt werden.

Als Alternative zu `xmodmap` gibt es das neuere Kommando `setxkbmap`. Es wird üblicherweise während des X-Startprozesses dazu verwendet, das gesamte Keyboard-Layout einzustellen (entsprechend der Konfiguration in `XF86Config`). Die zugrunde liegenden Dateien befinden sich in `/etc/X11/xkb` oder in `/usr/X11R6/lib/X11/xkb`.

12.3.12 Maus

12.3.13 Konfiguration in XF86Config

Die Maus wird wie die Tastatur durch einen `InputDevice`-Abschnitt konfiguriert, wobei als Treiber diesmal `Mouse` angegeben wird. Die folgenden Zeilen zeigen eine Minimalkonfiguration. Sie setzt voraus, dass die Maus über die Device-Datei (bzw. das Link) `/dev/mouse` angesprochen und eines der üblichen Protokolle verwendet wird.

```
Section "InputDevice"
  Identifier   "myMouse"
  Driver       "mouse"
  Option       "Protocol"    "Auto"
  Option       "Device"      "/dev/mouse"
EndSection
```

Zur Konfiguration der Maus sind folgende Schlüsselwörter vorgesehen:

- `Protocol` gibt an, wie die Kommunikation zwischen Maus und Computer erfolgt. Zur Auswahl stehen unter anderem folgende Varianten:

`Auto:`	XFree86 versucht das Protokoll selbst zu erkennen.
`ExplorerPS/2:`	Radmaus (kompatibel zu Microsoft IntelliMouse Explorer) an der PS/2-Schnittstelle
`IMPS/2:`	Microsoft-kompatible Radmaus an der PS/2-Schnittstelle
`IntelliMouse:`	Microsoft-kompatible Radmaus an der seriellen Schnittstelle
`PS/2:`	Standardmaus an der PS/2-Schnittstelle
`usb:`	USB-Maus

- `Device` gibt an, wie die Maus mit dem Computer verbunden ist. Üblich sind folgende Einstellungen:

`/dev/psaux:`	PS/2-Schnittstelle
`/dev/ttyS0:`	Erste serielle Schnittstelle
`/dev/ttyS1:`	Zweite serielle Schnittstelle
`/dev/usb/mouse0:`	USB-Schnittstelle

- `Buttons` gibt an, wie viele Tasten die Maus hat (per Default nimmt XFree86 an, dass es drei Tasten gibt). Beachten Sie, dass jedes Rad wie zwei Tasten gerechnet wird. Bei einer Maus mit drei Tasten und einem Rad lautet die richtige Einstellung also 5.
- `ZAxisMapping` gibt an, welchen virtuellen Buttons eventuell vorhandene Räder zugeordnet werden. Wenn Sie das Mausrad drehen, wertet XFree86 dies wie das Drücken eines Buttons aus. Jede Drehrichtung entspricht einem Button.)

  Die folgenden Ergänzungen im `InputDevice`-Abschnitt gelten für eine typische Radmaus mit einem Rad und zwei Maustasten (links und rechts). Allerdings kann das Rad selbst auch gedrückt werden, was der dritten Taste entspricht (die unter X ziemlich wichtig ist). Die Radbewegungen werden den virtuellen Tasten 4 und 5 zugeordnet.

  ```
  Option Buttons 5
  Option "ZAxisMapping" "4 5"
  ```

- `Emulate3Buttons` bewirkt, dass durch das gleichzeitige Drücken der rechten und linken Maustaste eine fehlende mittlere Maustaste simuliert werden kann. Mit `Emulate3Timeout` kann die Zeit in Millisekunden angegeben werden, innerhalb der beide Tasten gedrückt werden müssen (Empfehlung: 50 ms).

12.4 X.Org

Wie in der Einleitung dieses Kapitels bereits erwähnt wurde, hat XFree86 mit Version 4.4 die Lizenz des X-Servers verändert. Die neue Lizenz ist möglicherweise inkompatibel mit der GPL und daher auf den Unwillen der Linux-Entwicklergemeinde gestoßen. Fedora Core 2 war die erste große Distribution,

die aus diesem Grund auf den alternativen X-Server von X.Org umgestiegen ist. Später haben auch Mandriva (ab Version 10.1) und Suse (ab Version 9.2) und RHEL (ab Version 4) diesen Schritt vollzogen.

Im Folgenden werden ganz kurz die wichtigsten Änderungen zusammengefasst, die sich daraus ergeben. Dieser Abschnitt ist lediglich eine Ergänzung. Die meisten Informationen aus diesem Kapitel treffen weiterhin zu. Weitere Informationen zum neuen X-Server finden Sie unter *http://www.x.org/* sowie *http://freedesktop.org/ xorg/*.

Im Vergleich zu XFree86 haben sich einige Namen geändert:

Konfigurationsdatei:	`/etc/X11/XF86Config`	→	`/etc/X11/xorg.conf`
Server-Binary:	`/usr/X11R6/bin/XFree86`	→	`/usr/X11R6/bin/Xorg`
Logging-Dateien:	`/var/log/XFree86.n.log`	→	`/var/log/Xorg.n.log`

12.5 Schriftarten (Fonts)

Die Verwaltung der Schriftarten unter X ist ein relativ komplexes Thema, das wir in diesem Buch nicht vertiefen können. Im Laufe der letzten Jahre wurden sehr viele Erweiterungen am ursprünglichen X-Font-Konzept vorgenommen. Daraus ergeben sich aber mitunter widersprückliche Konsequenzen:

- X konnte ursprünglich nur mit nichtskalierbaren Fonts umgehen. Derartige Fonts stehen nur in bestimmten, vorgegebenen Größen zur Verfügung. Sie können zwar auch in anderen Größen angezeigt werden, wirken dann aber „pixelig“.
- Mittlerweile werden durch das `freetype`-Modul für XFree86 bzw. durch erweiterte Versionen des Font-Servers `xfs` auch moderne, frei skalierbare Fonts unterstützt (TrueType, Type-1).
- Manche Fonts stehen nur für bestimmte Zeichensätze zur Verfügung, z. B. für ISO-Latin-1. X unterstützt auch Unicode-Fonts (Zeichensatz ISO 10646-1). Wie viele Unicode-Fonts zur Verfügung stehen, können Sie mit dem Kommando `xlsfonts '*10646*'` testen.
- Für manche Schriftfamilien (z. B. Courier) können mehrere Fonts zur Auswahl stehen, die sich optisch nur in Details unterscheiden. Der Grund für diese Doppelgleisigkeit ist, dass Standardschriftfamilien von verschiedenen Herstellern bzw. Firmen implementiert wurden.
- Die Benennung von Fonts unter X ist sehr exakt, aber auch sperrig. Ein typischer Font-Name sieht etwa so aus:

  ```
  -adobe-new century schoolbook-bold-r-normal--8-80-75-75-p-56-
  iso8859-15
  ```

Die meisten modernen X-Programme (KDE, Gnome) verbergen diese Details vor den Anwendern. Den vollen Font-Namen müssen Sie nur bei der Konfiguration mancher alter X-Programme angeben (z. B. `xterm`).

- Manche Programme verwenden eine eigene Font-Verwaltung, die mit der von X nicht immer kompatibel ist. Das vielleicht wichtigste derartige Programm ist OpenOffice.
- Der Ausdruck von Dokumenten mit unterschiedlichen Fonts kann problematisch sein, weil das System für die Font-Verwaltung zur Bildschirmdarstellung (X, xfs) und das System für den Ausdruck (GhostScript) unterschiedlich sind.

Weitergehende Informationen zu Font-Grundlagen finden Sie im Font-HOWTO sowie auf der XFree86-Website:

http://www.tldp.org/HOWTO/Font-HOWTO/index.html
http://www.xfree86.org/current/fonts.html
http://www.tldp.org/HOWTO/FDU/index.html

12.6 X-Start (Window Manager)

Ähnlich wie beim Init-V-Prozess sind auch beim X-Startprozess viele Details distributionsabhängig. Im Folgenden finden Sie daher zuerst einen allgemeinen Abriss des X-Startprozesses und dann einige distributionsspezifische Details.

Sowohl für den Start von X als auch für das Benutzer-Login und den anschließenden Start von KDE, Gnome oder eines Window Managers ist ein so genannter Display-Manager zuständig. Wie so oft gibt es unter Linux nicht einen Display-Manager, sondern eine ganze Reihe. Der Urvater ist gewissermaßen der zu X gehörige `xdm`, populärer sind aber mittlerweile die KDE- und Gnome-Varianten `kdm` und `gdm`.

`xdm` ist das älteste der drei Programme, hat eine eher spartanische Benutzeroberfläche und bietet außer der eigentlichen Login-Funktion keinerlei Zusatzfunktionen. Dafür sind die Systemanforderungen (Bibliotheken, Speicherplatz etc.) bei weitem am geringsten.

`kdm` und `gdm` beglücken den Benutzer mit einer netten Oberfläche im Stil des jeweils dazugehörigen Desktops. Bei beiden Display-Managern ist es möglich, vor dem Login den gewünschten Desktop bzw. den Fenster-Manager auszuwählen (d. h. Sie können durchaus mit `kdm` Gnome bzw. mit `gdm` KDE starten!). Beide Programme ermöglichen es auch, den Rechner herunterzufahren (`shutdown`).

Der Display-Manager wird durch den Init-V-Prozess gestartet, wenn der Runlevel 5 aktiviert wird. Welcher Display-Manager verwendet wird, lässt sich konfigurieren (wobei die Details distributionsabhängig sind). Der weitere Verlauf

des X-Starts ist aber zum Glück vom eingesetzten Display-Manager weitgehend unabhängig.

Der Display-Manager startet als Erstes den X-Server. Zum Display-Manager gehören zahlreiche Konfigurations- und Script-Dateien. Sozusagen als zentrale Referenz, wo diese Dateien liegen, dient die Datei `xdm-config`. Diese befindet sich bei den meisten Distributionen im Verzeichnis `/etc/X11/xdm`. Der Display-Manager zeigt nun eine Login-Box an. Deren Aussehen kann je nach Display-Manager konfiguriert werden. Bei `kdm` und `gdm` kann beim Login der Name des gewünschten Desktops bzw. Window Managers angegeben werden. Dieser Name wird als Parameter an `Xsession` übergeben.

Nach dem Login wird das Script `Xsession` ausgeführt. Es kümmert sich darum, dass

- globale und lokale Änderungen am Tastaturlayout durchgeführt werden (`xmodmap`, `setxkbmap`),
- globale und lokale X-Ressourcendateien gelesen werden (`.Xdefaults`, `.Xresources`),
- alle Script-Dateien in einem Verzeichnis (z. B. `/etc/X11/xinit.d`) ausgeführt werden und schließlich
- der richtige Desktop oder Window Manager gestartet wird.

Je nach Konfiguration kann es auch sein, dass einige der hier beschriebenen Aufgaben von benutzerspezifischen Scripts (`~/.xinitrc` oder `~/.xsession`) übernommen werden. `Xsession` überprüft, ob derartige Scripts existieren, und verzweigt dann gegebenenfalls dorthin. In Wirklichkeit hat der `Xsession`-Startprozess noch mehr Optionen und Verästelungen. Außerdem kocht noch fast jede Distribution ihr eigenes Süppchen.

12.6.1 xdm-Konfiguration

Die zentrale `xdm`-Konfigurationsdatei heißt `xdm-config`. Sie befindet sich üblicherweise im Verzeichnis `/usr/X11R6/lib/X11/xdm` oder `/etc/X11/xdm`. Die einzige Aufgabe von `xdm-config` besteht darin, die Speicherorte einer ganzen Liste weiterer Konfigurations- und Script-Dateien anzugeben. Damit werden die diversen Teilaufgaben von `xdm` auf mehrere Scripts verteilt. Es fehlt hier der Platz, um alle `xdm`-Dateien zu beschreiben. Die folgende Liste enthält daher nur die wichtigsten `xdm-config`-Schlüsselwörter. In Klammern ist der übliche Dateiname der dazugehörigen Datei angegeben – dieser Name kann aber je nach Konfiguration variieren.

- `servers (Xservers)`: Dieses Script kümmert sich um den Start von X (noch bevor die Login-Box angezeigt wird).

- `setup (Xsetup` oder `Xsetup_0)`: Dieses Script kümmert sich um die Einstellung des `xdm`-Hintergrunds und eventuell um die Tastaturkonfiguration.
- `resources (Xresources)`: Diese X-Ressourcendatei steuert das Aussehen der `xdm`-Login-Box.
- `session (Xsession)`: Dieses Script ist für den Start des Window Managers und das Einlesen verschiedener benutzerspezifischer Konfigurationsdateien verantwortlich. Das Script ist stark distributionsabhängig.

12.6.2 kdm-Konfiguration

Der KDE-Display-Manager `kdm` ist weitgehend mit `xdm` kompatibel. Das bedeutet insbesondere, dass für den eigentlichen Start von X dieselben Dateien wie bei `xdm` verwendet werden. Das betrifft vor allem die oben beschriebene Konfigurationsdatei `xdm-config` sowie die meisten dort angegebenen Script-Dateien (hauptsächlich `Xsession`).

Die `kdm`-spezifischen Zusatzfunktionen (also die optische Gestaltung des Login-Dialogs, die Hintergrundgestaltung, die Darstellung der Benutzer durch Icons etc.) können von `root` im KDE-Kontrollzentrum (*Systemverwaltung* → *Anmeldungsmanager*) bequem eingestellt werden. Diese Einstellungen werden in der Datei `kdmrc` gespeichert. Wo sich diese Datei befindet, ist distributionsabhängig. Die Konfigurationsmöglichkeiten sind reichhaltig:

- Begrüßungstext, Begrüßungslogo, Dialogaussehen (GUI Style)
- Schriftart
- Hintergrundgestaltung (Hintergrund-Bitmap, Hintergrundprogramm)
- Informationen darüber, wer den Rechner herunterfahren darf
- Eine Liste mit den zur Auswahl stehenden Desktop-Systemen und Window Managern
- Eine Liste mit allen Benutzern, deren Namen beim Login durch einen Klick auf ein Icon ausgewählt werden können
- Auto-Login und Login ohne Passwort

`kdm` ermöglicht es, X per Mausklick ganz zu beenden (entspricht `init 3`) oder den Rechner herunterzufahren. Diese Möglichkeit können Sie entweder ganz sperren, nur `root` erlauben, allen (lokalen) Benutzern erlauben.

12.6.3 gdm-Konfiguration

`gdm` ist das Gnome-Gegenstück zu `kdm`. `gdm` kennt nicht ganz so viele Funktionen und Konfigurationsmöglichkeiten, das Prinzip ist aber dasselbe. Auch `gdm` berücksichtigt dieselben Konfigurationsdateien wie `xdm` (also `xdm-config`, `Xsetup_0` und `Xsession`, sowie alle Dateien, die von `Xsession` angesprochen werden). `gdm`-spezifische Einstellungen werden von `root` mit dem Programm `gdmconfig` durchgeführt. Die Einstellungen werden in `/etc/X11/gdm/gdm.conf` gespeichert. Die möglichen Desktop-Systeme werden als Script-Dateien in einem Verzeichnis angegeben. Der Ort des Verzeichnisses geht aus `gdm.conf` hervor.

Die Namen der Session-Dateien werden bei `gdm` in einem Menü zur Auswahl des gewünschten Desktops angezeigt. Nach einem Login wird die ausgewählte Session-Script-Datei einfach ausgeführt.

12.7 xterm und andere X-Utilities

Zusammen mit X werden einige einfache Programme mitgeliefert. Zu vielen dieser Programme gibt es mittlerweile modernere Versionen, die in KDE oder Gnome integriert sind, aber einige Utilities sind auch im Zeitalter moderner Desktops noch praktisch. Dieser Abschnitt stellt die wichtigsten Programme kurz vor.

12.7.1 xterm

`xterm` ist ein Shell-Fenster (Terminalfenster), in dem Kommandos ausgeführt werden können. Beim Start von `xterm` können diverse Optionen angegeben werden. Wichtig sind u. a.:

`-e kommando optionen`	führt in `xterm` das angegebene Programm aus. `-e` muss als letzte Option verwendet werden.
`-fn font`	gibt die gewünschte Schriftart an.
`-ls`	startet `xterm` mit einer Login-Shell.

Viele `xterm`-Einstellungen können Sie auch in einer Ressourcendatei vornehmen. Die folgenden Einstellungen bewirken, dass in `xterm` ein Scroll-Balken am rechten Fensterrand angezeigt wird und dass die letzten 1000 Zeilen in einem Puffer zwischengespeichert werden:

```
! in ~/.Xdefaults oder ~/.Xresources
xterm*scrollBar:       true
xterm*rightScrollBar:  yes
xterm*saveLines:       1000
```

Wenn im Shell-Fenster `xterm` die Sondertasten (Entf), (Backspace), (Pos1) und (Ende) nicht richtig funktionieren, fügen Sie die folgenden Zeilen in die Datei `~/.Xdefaults` ein:

```
! in ~/.Xdefaults oder ~/.Xresources
*VT100.Translations: #override \
       <KeyPress>BackSpace: string(0x7f)\n\
       <KeyPress>Delete: string(0x04)\n\
       <KeyPress>Home: string(0x01)\n\
       <KeyPress>End: string(0x05)
```

Hinweis

Neben `xterm` stehen unter X noch einige weitere Terminalprogramme zur Verfügung. Im Prinzip ist die Verwendung immer dieselbe, Unterschiede gibt es aber bei der Konfiguration, bei der Darstellung farbiger Texte (`ls`), bei den Tastenkürzeln, im Speicherverbrauch etc. Zur Auswahl stehen unter anderem `konsole` (KDE), `gnome-terminal`, `nxterm` und `rxvt`.

12.7.2 xhost

Eine Besonderheit von X besteht darin, dass Sie gleichzeitig unter verschiedenen Logins arbeiten können (also ohne sich vorher auszuloggen!). Das funktioniert auch im Netzwerkbetrieb. Sie können also ein Programm auf einem anderen Rechner ausführen: Dieses Programm wird aber am lokalen Rechner angezeigt und kann dort bedient werden.

Diese Möglichkeiten sind vor allem zur Administration praktisch: Wenn Sie als gewöhnlicher Benutzer eingeloggt sind und eine administrative Aufgabe erfüllen möchten, wechseln Sie einfach mit `su -l` in den Super-User-Modus (*root*-Login) und starten dann ein X-Programm (z. B. ein Konfigurationsprogramm oder einen Editor). Ebenso können Sie sich via `ssh` auf einem fremden Rechner einloggen und dort ein X-Programm starten. Sofern alle erforderlichen X-Bibliotheken zur Verfügung stehen, wird dieses Programm jetzt am *lokalen* Rechner angezeigt.

Damit dies funktioniert, müssen zwei Voraussetzungen erfüllt sein:

- Ihr Rechner muss die Anzeige eines X-Programms von einem fremden Rechner zulassen. Dazu muss auf dem lokalen Recher `xhost +rechnername` ausgeführt werden, wobei *rechnername* der Netzwerkname des externen Rechners ist.
- Nach dem Login am fremden Rechner müssen Sie die Variable `DISPLAY` so einstellen, dass sie den Namen Ihres eigenen Rechners enthält. Damit weiß das extern ausgeführte X-Programm, dass seine Oberfläche auf Ihrem lokalen Rechner erscheinen soll.

Wenn Sie mit `ssh` arbeiten und die Option `-X` verwenden, sind diese Voraussetzungen automatisch erfüllt. (Bei manchen Distributionen ist `ssh` so konfiguriert, dass diese Option per Default gilt.)

Wenn Sie dagegen einen Login mit `su` durchführen, müssen Sie sich je nach der Voreinstellung von X oft selbst um die `xhost`- und *DISPLAY*-Einstellungen kümmern. Die folgenden Kommandos illustrieren den Vorgang anhand eines Beispiels: Sie arbeiten als `jack` am lokalen Rechner `hades` und möchten nun per `vi` die Konfigurationsdatei `/etc/hosts` verändern:

```
jack@hades$  xhost +localhost
jack@hades$  su -l
Password:  ******
root@hades#  export DISPLAY=localhost:0
root@hades#  vi /etc/hosts &
root@hades#  exit
root@hades# logout
jack@hades$  xhost -localhost
```

Das abschließende Kommando `xhost -localhost` verbietet aus Sicherheitsgründen die weitere Nutzung von X durch andere Nutzer.

Das Programm `xkill` hilft dabei, hängen gebliebene oder halb abgestürzte X-Programme endgültig zu beenden. Die Bedienung ist denkbar einfach: Sie führen `xkill` aus und klicken dann mit der Maus das betreffende Fenster an.

`xlsfonts` zeigt eine Liste aller verfügbaren Zeichensätze an. Ähnlich wie bei `ls` kann die Liste durch ein Suchmuster eingegrenzt werden. Allerdings muss dieses Muster in Anführungszeichen gesetzt werden, damit es nicht von der `bash` ausgewertet wird.

Kapitel 13

Linux im lokalen Netz

Einführung

Dieses Kapitel beschreibt, wie ein Linux-Rechner in ein lokales Netzwerk (LAN) eingebunden wird. Sofern das lokale Netzwerk mit dem Internet verbunden ist, stehen damit auch alle Internet- und Netzwerkfunktionen auf Ihrem Rechner zur Verfügung. Wenn alles klappt, ist die LAN–Einbindung unter Linux in ein bis zwei Minuten erledigt. Oft erfolgt die Konfiguration schon während der Installation.
Dieses Kapitel geht davon aus, dass Sie Ihren Linux-Rechner an ein schon bestehendes lokales Netzwerk anschließen möchten. Falls Sie die Verbindung zum Internet selbst herstellen müssen, empfehlen wir generell die Anschaffung eines passenden Routers (ISDN, DSL etc.). Derartige Geräte sind inzwischen so preiswert geworden, dass die Kosten den Gewinn an Bequemlichkeit und vor allem Sicherheit bei weitem aufwiegen. Der Router wird gemäß der Hersteller- und Provideramgaben konfiguriert und ermöglicht dann jedem Rechner im Netz den Internetzugang. Gleichzeitig schützt er das Netz durch seine Firewall-Funktionen.

13.1 Loopback, IP-Adressen, Netzmaske

Was brauchen wir eigentlich alles, um einen Linux-Rechner ans Netz zu bringen? Eine Netzwerkkarte – klar! Die wird in der Regel schon bei der Installation erkannt und das passende Kernel-Modul eingebunden. Feststellen lässt sich das mit dem schon erwähnten Kommando `dmesg`. Falls Sie die Karte erst später eingebaut haben, finden Sie auf Seite 644 weitere Hinweise. Dann brauchen wir noch eine **IP-Adresse**, die **Netzmaske**, die **Netzwerkadresse** sowie **Broadcast-** und **Gateway**-Adresse.

Bei der derzeit aktuellen Version 4 des IP-Protokolls ist jede dieser Adressen eine 32-stellige Binärzahl. Weil aber nun 32 Nullen und Einsen etwas unübersichtlich

sind, hat man daraus einfach vier Gruppen zu je acht Stellen gemacht und schreibt diese vier Gruppen als Dezimalzahlen auf. Nachdem bei jeder Zahl Werte zwischen 0 und 255 auftreten können, schrebt man noch einen Punkt zwischen die Zahlen. Heraus kommt dabei z. B. 105.22.234.1 (was besser lesbar ist als binär 01101001000101101110101000000001).

Rechner in lokalen Netzen sind für das Internet im Regelfall unsichtbar. Das bedeutet aber nicht, dass die Rechner keine Internetfunktionen nutzen können. Aber diese Rechner sind vor unkontrollierten Zugriffen aus dem Internet geschützt. Rechner, die weltweit kommunizieren sollen, bekommen vom jeweiligen Internet Service Provider eine IP-Adresse zugeteilt. Im LAN ohne direkte Internet-Verbindung braucht man aber nur IP-Adressen, die im jeweiligen Netz eindeutig sein müssen, nicht aber weltweit. Es wurden daher im IP-Zahlenraum drei Bereiche für lokale Netzwerke reserviert, die man jederzeit verwenden darf:

10.0.0.0 – 10.255.255.255 (A-Netz)
172.16.0.0 – 172.31.255.255 (B-Netze)
192.168.0.0 – 192.168.255.255 (C-Netze)

Der erste Bereich ermöglicht theoretisch ein Netz mit 16 Millionen Rechnern – das reicht auch für sehr große Firmen. Beim zweiten Bereich handelt es sich um 16 Teilnetze mit je ca. 65 000 Adressen (z. B. 172.23.0.0 bis 172.23.255.255). Der dritte Bereich besteht aus 256 kleinen Teilnetzen mit jeweils 254 Adressen. Ganz egal, in welchem Teilnetz Sie Ihr lokales Netz bilden – es ist sichergestellt, dass es zu keinen Adresskonflikten mit „richtigen" IP-Internetadressen kommt.

Meist wollen Sie freilich auch innerhalb des lokalen Netzes Internetfunktionen nutzen (z. B. WWW-Seiten lesen). Um dies zu ermöglichen, muss innerhalb des lokalen Netzwerks ein Rechner bzw. der eingangs erwähnte Router als so genanntes Gateway zum Internet konfiguriert werden. Dieser Rechner/Router stellt die Verbindung zum Internet her (über DSL, ISDN, Modem etc.) und leitet alle Internetanforderungen des lokalen Netzes weiter. Das Gateway hat außerdem die Aufgabe, die lokalen IP-Adressen durch eine weltweit gültige IP-Adresse zu ersetzen. Sie suchen sich also ein (Teil-)Netz aus dem oben angegebenen Nummernkreis und vergeben daraus die Rechner-IP-Adressen.

Nun kommen Netzmaske, Netzwerk- und Broadcast-Adresse ins Spiel. Die Ausdehnung eines lokalen Netzes wird durch die Netzmaske eingeschränkt. Dabei handelt es sich abermals um vierteilige Zifferngruppen, die intern als Bitmuster für IP-Adressen verwendet werden. Die Netzmaske legt fest, welcher Teil der IP-Adresse die Netzwerkadresse ist und welcher die Rechneradresse innerhalb des Netzes (das sorgt nicht nur dafür, dass mehrere lokale IP-Netze dasselbe Kabel verwenden können, ohne sich gegenseitig zu stören, sondern es macht auch den Routern das Leben leichter. Die Netzmaske hat bei allen Stellen, welche die Netzwerkadresse repräsentieren, eine „1" stehen und dahinter lauter Nullen (z. B.: 11111111111111110000000000000000 entspricht 255.255.0.0). Die Netzwerkadresse hat ebenfalls an den Stellen, welche den Rechner (Host) repräsentieren, Nullen stehen. Die Broadcast-Adresse (Broadcast = Rundruf, an alle) hat beim Rechneranteil lauter Einsen. Dazu ein Beispiel:

Wenn das lokale Netz alle Nummern 192.168.12.*n* umfasst, lautet die dazugehörige Netzmaske 255.255.255.0, die Netzwerkadresse 192.168.12.0 und die Broadcast-Adresse 192.168.12.255. (Bei manchen Konfigurationsprogrammen brauchen Sie keine Netzwerkadresse anzugeben, da sich diese aus den beiden anderen Adressen ergibt.) Das resultierende Netzwerk wird jetzt mit 192.168.12.0/255.255.255.0 oder kurz mit 192.168.12.0/24 bezeichnet. (Die Kurzschreibweise gibt die Anzahl der binären Einser der Netzmaske an.) Zwei Rechner mit den IP-Adressen 192.168.12.71 und 192.168.12.72 können sich in diesem Netzwerk also direkt miteinander verständigen (weil die IP-Adressen im Bereich der Netzmaske übereinstimmen). Die maximale Anzahl von Rechnern, die gleichzeitig in diesem Netz kommunizieren können, beträgt 254 (.1 bis .254) – die Nummern .0 und .255 sind ja reserviert.

Ein Gateway ist ein Router oder Rechner, der an der Schnittstelle zwischen zwei Netzen steht (oft zwischen dem lokalen Netz und dem Internet). Damit Ihr Linux-Rechner in einem lokalen Netz auf das Internet zugreifen kann, muss bei der Konfiguration die Gateway-Adresse angegeben werden. Die Gateway-Adresse bezeichnet also einen Rechner im lokalen Netz – z. B. 192.168.12.254. Dieser Rechner hat insofern eine Sonderstellung, als er (z. B. per ADSL) mit dem Internet in Verbindung steht. Der Internetverkehr des gesamten lokalen Netzwerks erfolgt daher über den Gateway-Rechner.

Ein **Nameserver** ist ein Programm, das Rechnernamen bzw. Internetadressen (z. B. *www.yahoo.com*) in IP-Adressen übersetzt. Bei kleinen Netzen erfolgt die Zuordnung zwischen Namen und Nummern oft über eine Tabelle (Datei `/etc/hosts`). Im Internet übernehmen Rechner mit entsprechenden Datenbanken diese Aufgabe. Statt des Begriffs Nameserver ist auch die Abkürzung DNS für *Domain Name Server* oder *Services* üblich. Wenn Sie in einem Webbrowser die Site *www.yahoo.com* ansehen möchten, wird daher als Erstes der Nameserver kontaktiert, um die IP-Adresse des Webservers von *www.yahoo.com* herauszufinden. Erst nachdem das gelungen ist, wird eine Verbindung mit dieser IP-Adresse hergestellt.

Die Abkürzung **DHCP** steht für *Dynamic Host Configuration Protocol*. DHCP wird oft in lokalen Netzwerken verwendet, um die Administration des Netzwerks zu zentralisieren. Anstatt bei jedem Rechner getrennt die IP-Adresse, das Gateway, den Nameserver etc. einzustellen, wird der Router oder ein Rechner als DHCP-Server konfiguriert. Alle anderen Rechner im lokalen Netzwerk nehmen beim Systemstart Kontakt mit dem DHCP-Server auf und fragen diesen, welche Adressen und Einstellungen sie verwenden sollen. Damit reduziert sich die Client-Konfiguration auf ein Minimum.

Eine besondere Rolle spielt noch das **Loopback-Interface**: Diese Schnittstelle ermöglicht die Verwendung des Netzwerkprotokolls für lokale Dienste, also zur Kommunikation innerhalb des Rechners. Das klingt vielleicht widersinnig, ist aber für viele elementare Linux-Kommandos erforderlich. Der Grund: Manche Kommandos bauen ihre Kommunikation auf dem Netzwerkprotokoll auf, ganz egal, ob die Daten lokal auf dem Rechner bleiben oder über ein Netz

auf einem fremden Rechner weiterverarbeitet werden. Ein Beispiel dafür ist der Druckerdämon `lpd`, der das Spooling für den Drucker übernimmt und sowohl lokal als auch von fremden Rechnern genutzt werden kann.

Als IP-Adresse für das Loopback-Interface ist immer 127.0.0.1 vorgesehen. Alle Distributionen kümmern sich automatisch um die Konfiguration des Loopback-Interface, auch wenn ansonsten keine Netzwerkkonfiguration durchgeführt wird. Dieser Adresse ist in der Regel auch der Name „localhost" zugeordnet. Ob es funktioniert, können Sie ganz einfach testen:

```
user$  ping localhost
PING localhost (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=255 time=0.136 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=255 time=0.085 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=255 time=0.077 ms
--- localhost ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.077/0.099/0.136 ms
```

`ping` sendet Datenpakete an die angegebene Adresse und misst, wie lange es dauert, bis die Ankunft der Pakete bestätigt wird. `ping localhost` bzw. `ping 127.0.0.1` sollte auch dann funktionieren, wenn Ihr Rechner keine Netzwerkkarte besitzt!

Schließlich kann es sein, dass ein Rechner mit mehreren Netzwerkkarten ausgestattet ist. Jede dieser Karten gilt als eigenes Interface und hat daher eine eigene IP-Adresse. Auch die mit einem Modem oder einer ISDN-Karte hergestellte PPP-Verbindung zum Internet bildet ein Interface. Diesem Interface wird ebenfalls eine IP-Adresse zugeordnet, wobei diese Nummer im Regelfall vom Internet Provider bestimmt wird. (Es handelt sich um eine „dynamische IP-Adresse", weil bei jedem Login eine neue Adresse zugeteilt wird.)

Welche IP-Adresse sollen Sie verwenden?

- Ihr Rechner ist nicht Teil eines lokalen Netzes: Abgesehen von 127.0.0.1 für das Loopback-Interface benötigen Sie gar keine IP-Adresse. (Das gilt auch für den Fall, dass dieser Rechner später via Modem/ISDN/ADSL mit dem Internet verbunden wird.) Ihre einzige Konfigurationsaufgabe besteht darin, den Domain- und den Hostnamen anzugeben.

- Ihr Rechner ist Teil eines bestehenden lokalen Netzes: Die IP-Adresse muss sich innerhalb der gültigen Adressen für dieses Netzwerk befinden (z. B. 192.168.0.*) und sie muss darin eindeutig sein. Falls es im Netz einen DHCP-Server gibt, brauchen Sie keine IP-Adresse anzugeben, müssen aber festlegen, dass der Rechner als DHCP-Client arbeiten soll.

- Ihr Rechner soll ein lokales Netz gründen: Entscheiden Sie sich für einen privaten IP-Adressraum (z. B. 192.168.0.*) und weisen Sie dem Rechner eine IP-Adresse daraus zu.

13.2 Konfiguration von Netzwerkkarten

Wenn Sie bereits Netzwerkkenntnisse besitzen, finden Sie hier eine Kurzanleitung, wie Sie Ihren Rechner innerhalb weniger Minuten in das lokale Netz (*local area network = LAN*) einbetten. Dieser Abschnitt setzt voraus, dass entweder im Mainboard des Rechners ein LAN-Adapter integriert ist (was bei modernen Rechnern und fast allen Notebooks der Fall ist) oder dass Sie eine PCI- oder PCMCIA-Netzwerkkarte eingebaut bzw. eingesteckt haben.

Hintergrundinformationen zu den in diesem Abschnitt verwendeten Begriffen wie IP-Adresse, Gateway, Hostname etc. finden Sie in der im Anhang angegebenen Literatur oder unter *http://www.netzmafia.de/skripten/netze/*. Dort gibt es auch ein Kapitel zur Netzwerkverkabelung.

13.2.1 Netzwerkkonfiguration

Bei den meisten Distributionen gibt es schöne Tools zur Netzwerkkonfiguration, wenn nicht schon alles bei der Installation erledigt wird. Bei Red Hat bzw. Fedora ist das Programm `system-config-network` für die Netzwerkkonfiguration zuständig, bei Suse hilft natürlich wieder das YaST-Modul *Netzwerkgeräte → Netzwerkkarte*. Bei Debian ist leider manuelle Tätigkeit angesagt, wie wir sie im folgenden Abschnitt beschreiben. Zur Durchführung von Detaileinstellungen sehen die Programme meist mehrere Dialoge vor:

- Sie müssen auf jeden Fall **Rechnernamen** und **Nameserver** angeben. Optional kann der Hostname auch via DHCP eingestellt werden. Hier erfolgt auch die Nameserver-Konfiguration. Falls es einen DHCP-Server gibt, stellt dieser die IP-Adressen von mindestens einem Nameserver zur Verfügung. Nur wenn das nicht der Fall ist, müssen Sie die Adressen selbst eingeben.
- Im Routing-Dialog müssen Sie normalerweise nur die IP-Adresse des Internet-Gateways Ihres Netzwerks eingeben. Falls es einen DHCP-Server gibt, überträgt er diese Information und das Eingabefeld *Standard-Gateway* darf leer bleiben.

Nach dem Abschluss der Konfiguration wird die Netzwerkverbindung normalerweise sofort hergestellt. Nur bei PCMCIA-Karten kann es sein, dass Sie die Karte kurz entfernen und neu einstecken müssen.

Nun können Sie als `root` mit `ifconfig` testen, ob das Interface erfolgreich aktiviert wurde. Außerdem können Sie mit `ping` *ipadresse* testen, ob eine Netzwerkverbindung zu einem anderen Rechner im lokalen Netzwerk oder im Internet hergestellt werden kann. Schließlich können Sie mit `ping www.yahoo.com` testen, ob auch die DNS-Konfiguration klappt (d. h. ob die zu `www.yahoo.com` passende IP-Adresse gefunden wird). Wenn das funktioniert, müssen Sie nur noch einen beliebigen Webbrowser starten und schon können Sie durch das Internet surfen.

13.2.2 Manuelle Inbetriebnahme einer Netzwerkkarte

Um zu verstehen, was bei der Initialisierung der Netzwerkkarte vor sich geht, können Sie die Karte auch manuell aktivieren (bei Debian ist dies die normale Vorgehensweise). Im ersten Schritt müssen Sie sicherstellen, dass das richtige Kernel-Modul für Ihre Netzwerkkarte geladen wird. Das Problem besteht darin, dass es für unterschiedliche Karten unterschiedliche Module gibt und Linux von sich aus nicht selbstständig erkennt, welches Modul benötigt wird. Möglicherweise wissen Sie nicht, was für eine Netzwerkkarte Sie besitzen. Falls es sich um eine PCI-Karte handelt, gibt die virtuelle Datei `/proc/pci` Auskunft. Aus den folgenden Zeilen können Sie entnehmen, dass meine Karte den häufig verwendeten Realtek-Chip 8139 verwendet:

```
root#  cat /proc/pci
...
  Bus  0, device   9, function  0:
    Class 0200: PCI device 10ec:8139 (rev 16).
      IRQ 11.
      Master Capable.  Latency=32.  Min Gnt=32.Max Lat=64.
      I/O at 0xe000 [0xe0ff].
      Non-prefetchable 32 bit memory at 0xdb000000 [0xdb0000ff].
...
```

Der zweite Schritt besteht darin, der erkannten Karte den richtigen Treiber (siehe `/lib/modules/2.`*n*.*n*`/net/*` zuzuordnen. Besonders hilfreich ist dabei die Liste im Hardware-HOWTO unter:

http://www.tldp.org/HOWTO/Hardware-HOWTO/

Nun können Sie mit `modprobe` versuchen, das passende Modul manuell zu laden:

```
root#  modprobe 8139too
```

Wenn das funktioniert, können Sie die Netzwerkschnittstelle mit `ifconfig` aktivieren:

```
root#  ifconfig eth0 up
```

Wenn kein oder das falsche Kernel-Modul geladen ist, erscheint hier die Fehlermeldung *eth0: unknown interface: No such device*. Andernfalls haben Sie das richtige Modul gefunden. Damit dieses Modul in Zukunft automatisch geladen wird (also ohne `modprobe`-Kommando), müssen Sie die Zuordnung zwischen der Schnittstelle `eth0` und dem Kernel-Modul *8139too* in die Modulkonfigurationsdatei eintragen. Je nach Distribution und Kernel-Version gibt es hierfür unterschiedliche Orte, `/etc/modprobe.conf` oder `/etc/modules.conf`. Dort tragen Sie die Zeile „alias eth0 8139too“ ein. Linux kommt auch mit mehreren Netzwerkkarten gleichzeitig zurecht (sofern es zu keinen Konflikten mit den IO-Adressbereichen bzw. Interrupts kommt). Weitere Informationen zu diesem

Thema finden Sie im Hardware- und im Ethernet-HOWTO sowie im Internet unter `http://www.scyld.com/network/`.

Einen Sonderfall stellen PCMCIA-Netzwerkkarten dar. Für deren Erkennung ist das PCMCIA-Verwaltungsprogramm `cardmgr` verantwortlich. Die Zuordnung zu den richtigen Kernel-Modulen erfolgt durch die Datei `/etc/pcmcia/config`.

Um die Netzwerkkarte zu aktivieren, übergeben Sie den Namen der Schnittstelle (`eth0`) und die gewünschte IP-Adresse an `ifconfig`. Wenn Sie das Kommando anschließend ein zweites Mal ohne Adressangabe ausführen, zeigt es alle bekannten Informationen zur Netzwerkschnittstelle an:

```
root#  ifconfig eth0 192.168.0.2
root#  ifconfig eth0
eth0 Link encap:Ethernet  HWaddr 00:48:54:39:B0:26
     inet addr:192.168.0.2  Bcast:192.168.0.255  Mask:255.255.255.0
     inet6 addr: fe80::248:54ff:fe39:b026/64 Scope:Link
     UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
     RX packets:5 errors:0 dropped:0 overruns:0 frame:0
     TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:100
     RX bytes:368 (368.0 b)  TX bytes:664 (664.0 b)
     Interrupt:5 Base address:0xf00
```

Nun können Sie mit `ping` überprüfen, ob Sie Kontakt zu anderen Rechnern im lokalen Netzwerk aufnehmen können. Die Option `-c 2` bewirkt, dass genau zwei `ping`-Pakete versendet werden:

```
root#  ping -c 2 192.168.0.10
PING 192.168.0.10 (192.168.0.10) 56(84) bytes of data.
64 bytes from 192.168.0.10: icmp_seq=1 ttl=64 time=0.217 ms
64 bytes from 192.168.0.10: icmp_seq=2 ttl=64 time=0.177 ms
--- 192.168.0.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1003ms
rtt min/avg/max/mdev = 0.177/0.197/0.217/0.020 ms
```

Bei **Debian** werden die dauerhaften Netzwerkeinstellungen in der Datei `/etc/network/interfaces` getätigt. Diese Datei könnte folgendermaßen aussehen:

```
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

# The first network card
auto eth0
iface eth0 inet dhcp
```

Oben ist die Einstellung für DHCP vorgenommen worden. Rechner-IP usw. werden vom DHCP-Server bezogen. Soll der Rechner eine feste IP-Adresse bekommen, schreibt man in die Datei stattdessen:

```
...
# The first network card
auto eth0
iface eth0 inet static
address 192.168.0.2
netmask 255.255.255.0
gateway 192.168.0.254
```

Mit dem Kommando `ifup eth0` wird dann das Netz aktiviert (mit `ifdown eth0` können Sie das Ganze wieder stoppen).

Das Kommando `ping` funktioniert momentan nur, wenn Sie die richtige IP-Adresse angeben. Damit Sie stattdessen auch einen Rechnernamen angeben können, muss `/etc/resolv.conf` die IP-Adresse eines Nameservers enthalten. Das folgende Beispiel geht davon aus, dass es einen Nameserver im lokalen Netz (192.168.0.*) gibt. Der Nameserver kann aber auch außerhalb sein und vom Internet Provider zur Verfügung gestellt werden. (Es können auch mehrere Nameserver angegeben werden.)

```
# /etc/resolv.conv
nameserver 192.168.0.10
```

Das Kommando `ifup` setzt auch die Route für den Rechner selbst (das scheint auf den ersten Blick paradox, aber so wie mittels `ifconfig` eine Verbindung zwischen Ethernetkarte und IP-Adresse hergestellt wurde, wird mit der lokalen Route eine Verbindung zwischen Rechner und lokalem Netz hergestellt). Nun können Datenpakete innerhalb des lokalen Netzwerks versandt werden. Damit auch ein Kontakt nach außen (also in das Internet) möglich wird, muss der Rechner wissen, wohin er derartige Pakete leiten soll. Sie müssen dazu die Adresse des Internet-Gateways Ihres Netzwerks mit `route` angeben. Das folgende Beispiel geht davon aus, dass die IP-Adresse des Gateways 192.168.0.10 ist:

```
root# route add default gw 192.168.0.10
root# route -n
Kernel IP routing table
Destination Gateway      Genmask         Flags Metric Ref Use Iface
192.168.0.0 0.0.0.0      255.255.255.0   U     0      0     0 eth0
0.0.0.0     192.168.0.10 0.0.0.0         UG    0      0     0 eth0
```

Jetzt sollte es möglich sein, Pakete an beliebige Adressen im Internet zu senden:

```
root# ping -c 2 www.yahoo.com
PING www.yahoo.com (66.218.71.198) 56(84) bytes of data.
64 bytes from w1.rc.vip.scd.yahoo.com (66.218.71.198):
  icmp_seq=1 ttl=240 time=243 ms
```

```
64 bytes from w1.rc.vip.scd.yahoo.com (66.218.71.198):
  icmp_seq=2 ttl=240 time=233 ms
--- yahoo.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1005ms
rtt min/avg/max/mdev = 233.356/238.287/243.218/4.931 ms
```

Mit dem Kommando `ifup eth0` bei Debian wird auch gleich die Route zum Gateway gesetzt. Bein nächsten Hochfahren des Rechners geschieht das sowieso automatisch.

Um die Netzwerkkarte wieder zu deaktivieren, führen Sie `ifconfig eth0 down` bzw. unter Debian `ifdown eth0` aus:

13.2.3 Konfigurationsdateien und -Interna

Hinweis

Für alle Beispiele in diesem Abschnitt gilt: Der zu konfigurierende Rechner heißt `palermo`, seine Domain `mafia`. Andere Rechner im lokalen Netz heißen `catania`, `messina` etc. Das lokale Netz verwendet 192.168.0.*-Adressen. Der lokale Rechner hat die IP-Adresse 192.168.0.2. Der Gateway-Rechner im lokalen Netz hat die IP-Adresse 192.168.0.10. Am Gateway-Rechner läuft ein eigener Nameserver. Namen und Nummern haben natürlich nur Beispielcharakter.

Die Datei **/etc/hosts** enthält eine Liste der bekannten IP-Adressen und der zugeordneten Namen: 127.0.0.1 ist die Standard-IP-Adresse für das Loopback-Device. 192.168.0.2 ist die IP-Adresse, unter der der Rechner `palermo.mafia` im lokalen Netz erreichbar ist. Statt `palermo.mafia` ist auch der Kurzname `palermo` erlaubt.

```
# /etc/hosts
127.0.0.1     localhost               # Loopback-Interface des Rechners
192.168.0.2   palermo.mafia palermo   # IP-Adresse des lokalen Rechners
```

Bei Red Hat bzw. Fedora enthält die *localhost*-Zeile zusätzlich den Eintrag *localhost.localdomain*. Diese Zeile sollte unverändert bleiben

Wenn Sie die anderen Rechner im lokalen Netz ebenfalls namentlich ansprechen möchten, müssen Sie auch deren Namen in `/etc/hosts` angeben. Statt `ping 192.168.0.13` können Sie dann also einfach `ping messina` ausführen, um die Verbindung zum Rechner `messina` zu testen.

```
# /etc/hosts
127.0.0.1     localhost               # Loopback-Interface des Rechners
192.168.0.2   palermo.mafia  palermo  # IP-Adresse des lokalen Rechners
192.168.0.10  syrakus.mafia  syrakus  # IP-Adressen anderer Rechner
192.168.0.13  messina.mafia  messina  #   im LAN
...
```

Analoge Einträge sind natürlich in den `/etc/hosts`-Dateien aller Rechner im lokalen Netz erforderlich. Wenn es sich dabei um sehr viele Rechner handelt, wird die Administration der vielen `/etc/hosts`-Dateien zunehmend mühsam. Aus diesem Grund empfiehlt es sich bei größeren Netzwerken, auf einem Rechner einen Nameserver einzurichten. Dieser Rechner (also der Nameserver) weiß, wie alle anderen Rechner im Netzwerk heißen. Alle Rechner im lokalen Netz (also alle Clients) können den Nameserver kontaktieren, um diese Information zu ermitteln. `/etc/hosts` kann nun auf eine einzige Zeile für *localhost* reduziert werden. Allerdings muss `/etc/resolv.conf` richtig konfiguriert werden (siehe etwas weiter unten). Wenn gar keine Netzwerkanbindung besteht, kann die *localhost*-Zeile auch den lokalen Rechnernamen enthalten:

```
# /etc/hosts (wenn es kein lokales Netz gibt, nur Loopback)
127.0.0.1       localhost   palermo   palermo.mafia
```

Bei manchen Distributionen enthält `/etc/hosts` Adressen wie `::1` oder `fe00::0`. Das deutet darauf hin, dass dieser Teil der Netzwerkkonfiguration bereits mit dem zukünftigen Internetprotokoll IPv6 kompatibel ist.

Die Datei **/etc/host.conf** gibt an, wie TCP/IP unbekannte IP-Adressen ermitteln soll. Die folgende Beispieldatei bestimmt, dass zuerst die Datei `/etc/hosts` ausgewertet (Schlüsselwort *hosts*) und danach der in `/etc/resolv.conf` angegebene Nameserver befragt werden soll (*bind*). Die zweite Zeile erlaubt, dass einem in `/etc/hosts` angegebenen Hostnamen mehrere IP-Adressen zugeordnet werden dürfen. Diese Datei liegt bei fast allen Distributionen in der hier angegebenen Form vor und muss nicht verändert werden.

```
# /etc/host.conf
order hosts, bind
multi on
```

Die Datei **/etc/resolv.conf** steuert, wie die IP-Adressen für unbekannte Netzwerknamen (Hostnamen) ermittelt werden. Unbekannt bedeutet, dass die Namen nicht in `hosts` definiert sind. Mit den Schlüsselwörtern *domain* und *search* wird erreicht, dass unvollständige Namen (etwa *catania*) mit dem Domain-Namen erweitert werden (zu *catania.mafia*). Das erhöht in erster Linie die Bequemlichkeit, weil lokale Internetnamen in verkürzter Form angegeben werden können. Bei `search` dürfen mehrere Domain-Namen angegeben werden (bei `domain` nur einer); dafür hat der `domain`-Name Vorrang vor den `search`-Namen, wird also zuerst getestet. Wenn wie hier nur ein einziger Domainname angegeben wird, kann auf die `domain`-Zeile verzichtet werden.

Die wichtigsten Einträge in `/etc/resolv.conf` werden mit dem Schlüsselwort *nameserver* eingeleitet: Damit können bis zu drei IP-Adressen von Nameservern angegeben werden. Diese Server werden immer dann angesprochen, wenn die IP-Adresse eines unbekannten Rechnernamens ermittelt werden soll. Die Angabe eines Nameservers ist daher unbedingt erforderlich, damit Internetadressen in

IP-Adressen aufgelöst werden können. (Als Privatanwender erhalten Sie die IP-Adresse eines DNS von Ihrem Internet Service Provider. In größeren lokalen Netzen gibt es zumeist eigene Nameserver direkt im lokalen Netz.)

```
# /etc/resolv.conf
domain mafia                # Hostnamen gelten für .mafia
search mafia                # Hostnamen gelten für .mafia
nameserver 192.92.138.35    # erster DNS
nameserver 195.3.96.67      # zweiter DNS (falls der erste ausfällt)
```

Falls es in Ihrem Netzwerk einen lokalen Nameserver gibt, geben Sie entsprechend dessen IP-Adresse an. Je nach Netzwerkkonfiguration wird `resolv.conf` dynamisch erzeugt. Wird eine Internetverbindung per PPP (Modem, ISDN oder ADSL) hergestellt, trägt das Programm für den Verbindungsaufbau automatisch die *nameserver*-Adressen Ihres Internet-Providers in `/etc/resolv.conv` ein. Ist Ihre lokale Netzwerkverbindung (LAN, WLAN) mit DHCP konfiguriert, trägt das Verbindungs-Programm die vom DHCP-Server übertragenen Nameserver-Adressen ein.

Die automatische Anpassung von `resolv.conf` ist in den meisten Fällen zweckmäßig. Wenn Sie dies aber nicht wünschen, können Sie die automatische Veränderung verhindern. Bei Red Hat bzw. Fedora müssen Sie dabei in die Datei für die jeweilige Schnittstelle (`/etc/sysconfig/network-scripts/ifcfg-xxxx`) die Zeile „PEERDNS=no" eintragen.

Bei Suse sind zwei Konfigurationsdateien betroffen:

```
# /etc/sysconfig/network/config für Modem, ISDN, ADSL (Suse)
MODIFY_RESOLV_CONF_DYNAMICALLY="no"

# in /etc/sysconfig/network/dhcp für DHCP-Clients (Suse)
DHCLIENT_MODIFY_RESOLV_CONF="no"
```

Bei Debian gilt nur, was Sie selbst in die Datei `resolv.conf` schreiben.

Für die **Gateway-Konfiguration** gibt es leider keinen einheitlichen Standard. In lokalen Netzen wird die Adresse des Gateways meist per DHCP übermittelt. Bei einer statischen Konfiguration sind je nach Distribution unterschiedliche Dateien verantwortlich. Bei Red Hat bzw. Fedora enthält die oben genannte Konfigurationsdatei für die Netzwerkschnittstelle die Variable `GATEWAY`. Sie tragen also beispielsweise `GATEWAY=192.168.0.10` ein.

Bei Suse erfolgt die Konfiguration zentral durch die Datei `/etc/sysconfig/network/routes`. Hier müssen Sie `default 192.168.0.10 - -` eintragen.

Der aktuelle **Hostname** kann mit dem Kommando `hostname` ermittelt werden. Soweit der Hostname nicht durch DHCP eingestellt wird, erfolgt die Konfiguration bei Red Hat in der Datei `/etc/sysconfig/network`. Bei Suse ist hingegen die Datei `/etc/HOSTNAME` und bei Debian `/etc/hostname` verantwortlich. Beachten Sie, dass Sie in allen Fällen auch `/etc/hosts` anpassen müssen, falls der Hostname ebenfalls in dieser Datei enthalten ist.

Die Netzwerkkarten, die in die Rechner eingebaut sind, werden beim Hochfahren des Systems durch das Script `/etc/init.d/network` initialisiert, das seinerseits eine Menge anderer Scripts und Kommandos aufruft und diverse Konfigurationsdateien auswertet.

Die Initialisierung von PCMCIA-Netzwerkkarten erfolgt durch das Hintergrundprogramm `cardmgr`, das auf Notebooks läuft. Wenn dieses Programm erkennt, dass eine Netzwerkkarte eingesteckt wurde, werden nach deren Hardware-Initialisierung dieselben Scripts aufgerufen wie von `/etc/init.d/network`.

Das Script */etc/init.d/network* wird während des Rechnerstarts im Rahmen des Init-V-Prozesses ausgeführt (siehe auch Seite 604). Es startet zuerst die Loopback-Schnittstelle für *localhost* und dann alle bekannten Netzwerkschnittstellen. Dazu wird für jede einzelne Schnittstelle `ifup` aufgerufen.

Nach Änderungen an der Netzwerkkonfiguration ist es oft empfehlenswert, mittels `/etc/init.d/network restart` die Netzwerkfunktionen neu zu starten. (Wenn Sie unter KDE oder Gnome arbeiten, sollten Sie sich anschließend aus- und dann neu einloggen.)

13.2.4 Netzwerkfunktionen testen

Dieser Abschnitt fasst nochmals die Kommandos zusammen, mit denen Sie verschiedene Funktionen der Netzwerkkonfiguration testen können. Eine genauere Beschreibung der Kommandos finden Sie auf Seite 644, wo diese Kommandos zur manuellen Konfiguration der Netzwerkkarte vorgestellt wurden.

Wenn Sie `ifconfig` ohne Parameter ausführen, werden alle bekannten Netzwerkschnittstellen aufgelistet. Für das Loopback- und ein Ethernet-Interface sollte das Resultat etwa so aussehen:

```
root# ifconfig
eth0 Link encap:Ethernet  HWaddr 00:48:54:39:B0:26
     inet addr:192.168.0.2  Bcast:192.168.0.255  Mask:255.255.255.0
     inet6 addr: fe80::248:54ff:fe39:b026/64 Scope:Link
     UP BROADCAST NOTRAILERS RUNNING MULTICAST  MTU:1500  Metric:1
     RX packets:12163 errors:0 dropped:0 overruns:0 frame:0
     TX packets:10850 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:100
     RX bytes:8528525 (8.1 Mb)  TX bytes:1909913 (1.8 Mb)
     Interrupt:5 Base address:0xf00
lo   Link encap:Local Loopback
     inet addr:127.0.0.1  Mask:255.0.0.0
     inet6 addr: ::1/128 Scope:Host
     UP LOOPBACK RUNNING  MTU:16436  Metric:1
     RX packets:177 errors:0 dropped:0 overruns:0 frame:0
     TX packets:177 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:0
     RX bytes:12446 (12.1 Kb)  TX bytes:12446 (12.1 Kb)
```

Wenn die Schnittstelle *eth0* fehlt, wurde die Netzwerkkarte noch gar nicht aktiviert. Versuchen Sie eine manuelle Aktivierung durch `ifconfig eth0 192.168.0.2`. Wenn dabei der Fehler *eth0: unknown interface: No such device* auftritt, fehlt dem Kernel das Modul zur Ansteuerung der Netzwerkkarte. Abhilfe schafft ein Eintrag in die Modulkonfigurationsdatei.

Mit `ping` können Sie nun in mehreren Stufen diverse Netzwerkfunktionen testen. `ping localhost` testet, ob das Loopback-Interface (und damit die elementaren Netzwerkfunktionen) zufriedenstellend arbeitet. Das sollte selbst dann gehen, wenn Sie gar keine Netzwerkkarte haben. Indem Sie statt `localhost` die IP-Adresse eines anderen Rechners im lokalen Netz angeben, testen Sie, ob das lokale Netz funktioniert:

```
root#  ping -c 2 192.168.0.3
PING 192.168.0.99 (192.168.0.3): 56 data bytes
64 bytes from 192.168.0.99: icmp_seq=0 ttl=255 time=0.274 ms
64 bytes from 192.168.0.99: icmp_seq=1 ttl=255 time=0.150 ms
--- 192.168.0.99 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.150/0.212/0.274 ms
```

Wenn Sie dem Rechner mit der IP-Adresse 192.168.0.3 in `/etc/hosts` einen Namen gegeben haben oder wenn es im Netz einen Nameserver gibt, können Sie bei `ping` statt der IP-Adresse den Rechnernamen angeben.

Als Nächstes können Sie testen, ob die Verbindung zum Internet gelingt. Das folgende Kommando testet gleichzeitig zwei Aspekte der Netzwerkkonfiguration: die Erreichbarkeit des Nameservers und die Funktion des Gateways.

```
root#  ping www.berliner-kaffeeroesterei.de
PING www.berliner-kaffeeroesterei.de (195.167.223.181): 56 data bytes
64 bytes from 195.167.223.181: icmp_seq=0 ttl=239 time=70.860 ms
64 bytes from 195.167.223.181: icmp_seq=1 ttl=239 time=68.709 ms
...
(Abbruch mit Strg + C)
```

Wenn das nicht funktioniert, sind mehrere Ursachen denkbar:

- Vielleicht ist der Server der Berliner Kaffeerösterei gerade unerreichbar oder der Server hat aus Sicherheitsgründen die Antwort auf `ping` deaktiviert. Probieren Sie einfach eine andere bekannte Internetadresse aus.
- Wenn der Nameserver nicht funktioniert, erhalten Sie die Fehlermeldung *ping: unknown host www.berliner-kaffeeroesterei.de*. Überprüfen Sie, ob `/etc/resolv.conf` die Adresse des Nameservers enthält.
- Wenn das Gateway nicht funktioniert, erhalten Sie die Fehlermeldung *connect: Network is unreachable*. Führen Sie das Kommando `route -n` aus. Die letzte Zeile sollte so wie beim folgenden Beispiel aussehen:

```
root# route -n
Kernel IP routing table
Destination  Gateway      Genmask       Flags Metric Ref Use Iface
192.168.0.0  0.0.0.0      255.255.255.0 U     0      0   0   eth0
127.0.0.0    0.0.0.0      255.0.0.0     U     0      0   0   lo
0.0.0.0      192.168.0.10 0.0.0.0       UG    0      0   0   eth0
```

- Falls Sie das Gateway selbst eingerichtet haben, besteht die Möglichkeit, dass Sie die Masquerading-Funktion vergessen haben. In diesem Fall würde der Internetzugang für das gesamte lokale Netzwerk nicht funktionieren.

Anhang A

Literatur

Einführung Diese Literaturliste bietet eine kleine Auswahl aus einer großen Zahl von Büchern zur Vertiefung der einzelnen Kapitel. Außerdem finden Sie hier Literatur zu Programmen oder Themen, die im Buch nur knapp behandelt werden konnten.

A.1 Linux und Unix

1. Michael Kofler:
   Linux
   Verlag Addison Wesley

2. J. Gulbins et al.:
   UNIX
   Springer Verlag

3. Stefanie Teufel:
   Jetzt lerne ich SuSE Linux
   Verlag Markt & Technik

4. David Pitts, Bill Ball:
   Linux Kompendium
   Verlag Markt & Technik

5. Jessica Heckman:
   Linux in a Nutshell

Verlag O'Reilly

6. Michael Renner:
   Linux für Onliner
   Verlag O'Reilly

7. Jessica Perry Hekman:
   Linux in a Nutshell
   Verlag O'Reilly

8. Nemeth/Snyder/Seebass:
   Systemadministration unter UNIX
   Verlag Prentice-Hall

9. Aeleen Frisch:
   Essential System Administration
   Verlag O'Reilly

   Helmut Herold:
   UNIX-Grundlagen
   Verlag Addison-Wesley

10. Helmut Herold:
    UNIX-Shells
    Verlag Addison-Wesley

11. Rainer Krienke:
    UNIX Shell-Programmierung
    Hanser-Verlag

12. Wolfgang Mauerer:
    Linux Kernelarchitektur
    Hanser-Verlag

13. Garfinkel/Spafford:
    Practical UNIX Security
    Verlag O'Reilly

A.2 LaTeX

1. Wolfgang Mauerer:
   Textverarbeitung mit LaTeX 2_ε
   Hanser-Verlag

2. Helmut Kopka:
   LaTeX-Einführung, Band 1 - 3
   Verlag Addison-Wesley

3. Michael Goosens, Frank Mittelbach, Alexander Samarin:
   Der LaTeX-Begleiter
   Verlag Addison-Wesley

4. Karsten Günther:
   LaTeX ge-packt
   Verlag mitp

A.3 Anwendungsprogramme

1. Ramin Assisi:
   Open Office Für Ein und Umsteiger
   Hanser-Verlag

2. Kai Surendorf:
   Wissenschaftliche Arbeiten mit OpenOffice.org 2.0
   Galileo Press

3. Tobias Berndt:
   OpenOffice.org
   Galileo Press

4. Jacqueline Rahemipour:
   Textverarbeitung mit OpenOffice.org 2.0
   Galileo Press

5. Jörg Schmidt:
   Tabellenkalkulation mit OpenOffice.org 2.0 Calc
   Galileo Press

6. Jürgen Osterberg:
GIMP 2
dpunkt.verlag

7. Klaus Gölker:
Fotobearbeitung und Bildgestaltung mit dem GIMP 2
dpunkt.verlag

8. Michael Kofler:
MySQL 5 - Einführung, Anwendung, Referenz
Verlag Addison-Wesley

9. Paul Dubois:
MySQL Kochbuch
Verlag O'Reilly

A.4 Programmieren in C, C++

1. Goll, Bröckl, Dausmann:
C als erste Programmiersprache
Teubner-Verlag

2. Karlheinz Zeiner:
Programmieren lernen mit C
Hanser-Verlag

3. Henning Mittelbach:
Einführung in C
Fachbuchverlag Leipzig

4. B. W. Kernighan and D. M. Ritchie:
Programmieren in C
Hanser-Verlag

5. C. L. Tondo/S. E. Gimpel:
Das C-Lösungsbuch zu K. & R.
Hanser-Verlag

6. A.Böttcher, F. Kneißl:
Informatik für Ingenieure
Oldenbourg-Verlag

7. Nancy Richter, Ronny Kissing:
KDE-Entwicklung mit KDevelop 3.x
Bomots Verlag

A.5 Programmieren in Perl

1. R. L. Schwartz:
Einführung in Perl
Verlag O'Reilly

2. Larry Wall, T. Christiansen, R. L. Schwartz:
Programmieren mit Perl
Verlag O'Reilly

3. T. Christiansen, N. Torkington:
Das Perl-Kochbuch
Verlag O'Reilly

4. N. Patwardhan, C. Irving:
Programmieren mit Perl-Modulen
Verlag O'Reilly

5. Rainer Krienke:
Programmieren in Perl
Hanser Verlag

6. Laura Lemay:
Perl in 21 Tagen
Verlag Markt & Technik

7. Farid Hajji:
Perl - Einführung, Anwendung, Referenz
Verlag Addison-Wesley

8. John Orwant et. al.:
   Algorithmen mit Perl
   Verlag O'Reilly

A.6 Administration und Netzwerk

1. Evi Nemeth, Garth Snyder et al.:
   Handbuch zur UNIX Systemverwaltung
   Verlag Markt&Technik

2. Jochen Hein:
   Linux Systemadministration
   Verlag Addison Wesley

3. Olaf Kirch:
   Linux Netzwerkadminstation
   Verlag O'Reilly

4. S. Fischer, U. Walther:
   Linux Netzwerke
   Suse Press/Millin

5. Jörg Holzmann, Jürgen Plate:
   Linux-Server für Intranet und Internet
   Hanser Verlag

6. Anonymous:
   Der neue Linux Hackers Guide
   Markt & Technik

A.7 Allgemeine Informationen

1. Linus Torvalds:
   Just for Fun
   Hanser Verlag

2. Jens Sieler-Hornke:
   Kommunizieren unter Linux
   Hanser Verlag

3. Florian Schiel:
   BAfH Bastard Assistent from Hell
   Schwarten
   oder unter *http://bofh.ntk.net/Bastard.html*

4. Walter Moers:
   Die 13 1/2 Leben des Käpt'n Blaubär
   Eichborn Verlag

Anhang B

Die DVDs zum Buch

Einführung

Die beiden DVDs im Buch enthalten eine komplette Debian Linux-Distribution (Version: Debian GNU/Linux 3.1r1 „Sarge“), die Sie auf Ihrem Rechner installieren können. Zur Installation muss das DVD-Laufwerk im BIOS-Setup des Rechners als Bootlaufwerk eingestellt bzw. freigegeben sein (meist heist der entsprechende Menüpunkt „Bootreihenfolge ändern“). Die Installation beginnen Sie, indem Sie die DVD 1 in das DVD-Laufwerk Ihres PC stecken und den Rechner neu starten.

Sollten Sie bereits eine Linux-Distribution installiert haben, sind Sie mit den grundsätzlichen Schritten einer Linux-Installation vertraut. Debian unterscheidet sich bei der Installation kaum von anderen Distributionen:

Ein wesentlicher Unterschied gegenüber anderen Linux-Distributionen ist, dass Debian GNU/Linux während der Installation nicht über eine grafische Benutzeroberfläche konfiguriert wird. Alle Anpassungen sind über die Tastatur einzugeben, denn Debian ist auch für ältere Hardwareplattformen verfügbar, die für eine grafische Benutzeroberfläche eventuell zu schwach auf der Brust sind. Andererseits gehört es zur Philosophie von Debian, nur die unbedingt notwendigen Komponenten zu installiert.

Wenn diese Kurzeinweisung nicht ausreichen sollte, lesen Sie das Kapitel „Debian 3.1 (Sarge) Installation“ im Debian Anwenderhandbuch nach, das Sie im Web unter *http://debiananwenderhandbuch.de/* finden. Dort wird ausführlicher auf die Installation eingegangen. Diese Anleitung soll fortgeschrittenen-Anwendern die Möglichkeit geben, relativ schnell zu einem lauffähigen System zu kommen.

- Sie benötigen eine freie oder nicht mehr benötigte Partition oder – noch besser – eine unbenutzte Festplatte. Haben Sie keine Partition mehr frei, können Sie mit dem Programm `fips.exe` (auf der CD im Verzeichnis `install`), um eine bestehende Partition unter DOS zu verkleinern.

- Booten Sie nun direkt von der DVD oder erstellen Sie Bootdisketten aus den Dateien `boot.img`, `root.img`, `cd-drivers.img` und `net-drivers.img`, die sich alle im Verzeichnis `/install/floppy/` auf der CD-ROM befinden. Benutzen Sie dazu das Linux-Kommando `dd if=boot.img of=/dev/fd0` oder das DOS-Programm `rawrite2.exe`.
- Beantworten Sie alle Fragen des Debian-Installationsprogramms mit der Eingabe-/Returntaste, wenn Sie sich mit der Antwort nicht sicher sind. Unter anderem sind dies, die Wahl der gewünschten Tastaturbelegung Und des Landes sowie die Einteilung des freien Festplattenplatzes in mindestens zwei Partitionen (eine als Swap- und eine Root-Partition).
- Installieren Sie nun das Grundsystem von der DVD.
- Starten Sie das System neu, und folgen Sie den Anweisungen zur abschließenden Konfiguration des Systems. In diesem Schritt können auch weitere Softwarepakete auf dem System installiert werden.

Tipp

Sollten Sie schon ein Linux-System verwendet haben und nun Debian neu darüber installieren, empfiehlt es sich, wichtige Konfigurationsdateien zu sichern (am besten das ganze Verzeichnis `/etc`) oder auszudrucken. Auch ist es günstig, sich die X-Konfiguration `XF86Config*` zu sichern. Dann geht die Einrichtung von Diensten und die der grafischen Oberfläche nachher schneller. Mit dem Kommando `lspci` erfahren Sie auch, welche Basishardware installiert ist.

Achten Sie besonders bei der Festplattenpartition darauf, dass Sie die richtige Platte bzw. Partition verwenden und nicht aus Versehen Ihren PC komplett platt machen. Als Dateisystem empfehlen wir Ihnen *ext3* oder *reiser*.

Abschließend wird ein Linux-Kernel auf dem System installiert. Falls auf dem Installationsmedium verschiedene Kernel-Pakete vorhanden sind, können Sie jetzt einen geeigneten Kernel auswählen. Verwenden Sie an dieser Stelle möglichst immer den Kernel, der auch für die Installation verwendet wurde. Wenn mit diesem Kernel die Installation bis zu dieser Stelle reibungslos geklappt hat, wird das System auch nach dem Neustart damit einwandfrei laufen. Sie können später immer noch andere Kernel-Pakete nachinstallieren.

Im letzten Schritt wird der GRUB-Bootloader installiert. Es wird auf allen Festplatten und Partitionen nach bereits auf dem System installierten anderen Betriebssystemen gesucht und diese automatisch in die Konfiguration des Bootloaders aufgenommen.

Nach dem einzigen Neustart des Systems wird das Grundsystem konfiguriert. Die durchgeführten Schritte können jederzeit durch den Aufruf des Programms `base-config` wiederholt (und so geändert) werden.

Die Basisinstallation von Debian sieht keine grafische Benutzeroberfläche vor. Ungeachtet dessen, ist es natürlich ohne weiteres möglich, auch mit Debian

zu einer grafischen Benutzeroberfläche zu kommen. Die aktuelle Version 4 von XFree86 ist ab Debian GNU/Linux 3.0 in die Distribution integriert worden. Alle benötigten Pakete für die Grafikoberfläche können Sie mittels

```
apt-get install xserver-xfree86 xserver-common xfonts-base \
gnome-core gnome-panel gnome-control-center sawfish-gnome \
gnome-session gnome-terminal
```

installieren. In Kapitel 10 des Debian Anwenderhandbuchs ist die Vorgehensweise genau beschrieben, Sie brauchen aber in der Regel auch hier nur das `dbconf`-Programm.

Näheres zur Linux-Konfiguration finden Sie in Kapitel 11 und zur X-Konfiguration in Kapitel 12 in diesem Buch.

Anhang C

Lösungen der Aufgaben

Einführung

In diesem Kapitel des Anhangs sind Lösungsvorschläge zu den Aufgaben der Shell-, LaTeX- und Programmier-Kapitel abgedruckt. Wenn Ihre Lösung nicht genau so aussieht, heißt das aber nicht, dass diese unbedingt falsch sein muss. Gerade bei der Programmierung und auch bei LaTeX führen viele Wege zum Ziel.

C.1 Shell

1. Es geht um Kommandosubstitution. Was beinhaltet nach Abarbeitung der Kommandofolge die Datei `datei3`?

```
echo "blau" > kunde
echo "gruen" > kun.de
echo "gelb" >> kunde
echo "rot" > kunde.r
echo kund* >datei2
cat `cat datei2` > datei3
```

Lösung

```
echo "blau" > kunde        # blau

echo "gruen" > kun.de      # gruen

echo "gelb" >> kunde       # blau
                           # gelb

echo "rot" > kunde.r       # rot

echo kund* >datei2         # kunde
                           # kunde.r

cat ‘cat datei2‘ > datei3  # blau
                           # gelb
                           # rot
```

2. Mit Pipes kann man recht komfortable Kommandos bilden. Verwenden Sie die Hintereinanderschaltung von `ps aux` und `grep`, um die Prozesse eine bestimmten Benutzers (z. B. root) oder eines bestimmten Terminals anzuzeigen.

Lösung

```
# zum Beispiel:
user$ ps aux | grep sshd
root   316  0.0  0.3  2896 1300 ?   Ss  09:23  0:00 /usr/sbin/sshd
root   453  0.0  0.4  5912 1768 ?   S   09:36  0:00 /usr/sbin/sshd
plate  455  0.0  0.5  5920 1880 ?   S   09:36  0:00 /usr/sbin/sshdpe
```

3. Geben Sie einen Aufruf von `find` an, der im Verzeichnis `/home` alle Dateien sucht, auf die innerhalb der letzten 10 Tage zugegriffen wurde.

Lösung

```
find /home -mtime -10 2>/dev/null
# die Umleitung der Fehlerausgabe beseitigt Fehlermeldungen,
# wenn man für das entsprechende Verzeichnis keine
# Zugriffsrechte besitzt
```

4. Welches `find`-Kommando müsste man verwenden, um in allen Dateien auf der Platte mit der Endung „.txt“ nach der Zeichenkette „UNIX“ zu suchen.

Lösung

```
find / -name '*.txt' | xargs grep "UNIX"
```

5. Schreiben Sie ein Shell-Skript, das in einer Endlosschleife alle Sekunden die Uhrzeit (Stunde, Minute und Sekunde) in der linken oberen Ecke des Bildschirms ausgibt.

Lösung

```
while :
  do
  tput home
  date "+[%H:%M:%S]"
  sleep 1
  done
```

6. Schreiben Sie ein Shell-Skript „ggt“, daß den größten gemeinsamen Teiler der als Parameter übergebenen beiden Zahlen berechnet. Formulieren Sie die Berechnung des GGT als Shell-Funktion. Der Algorithmus lautet folgendermaßen:

```
ggt(x,y):
solange x ungleich y ist, wiederhole
  falls x > y  dann  x = x - y
  sonst              y = y - x;
```

Denken Sie daran, daß man zum Rechnen das Kommando `expr` braucht.

Lösung

```
x=$1
y=$2
while [ $x -ne $y ]
  do
  if [ $x -gt $y ]; then
    x=`expr $x - $y`
  else
    y=`expr $y - $x`
  fi
  done
echo $x
```

7. Eine Datei namens `personal` habe folgendes Aussehen:

```
[Name]        [Vorname]     [Wohnort]     [Geb.-Datum]
=======================================================
Meyer         Peter         Berlin         10.10.1970
Schulze       Axel          Hamburg        12.12.1980
Lehmann       Rita          München        17.04.1971
```

Sortieren Sie die Daten der Datei (ohne Zeile 1 und 2) nach dem Namen und geben Sie das Ergebnis in eine Datei `personal.sort` aus. Die beiden Überschriftszeilen sollen natürlich wieder am Dateianfang stehen.

Lösung

```
head -2 personal > hilf
cat personal | sed -e '1,2d' | sort >> hilf
mv hilf personal
```

8. Schreiben Sie ein Shellscript, das an alle Benutzer mit der Gruppennummer 100 eine E-Mail verschickt. Betreff und die Datei, welche den E-Mailtext enthält, werden als Parameter an das Script übergeben.

Lösung

```
Subject=$1
File=$2
# Hier waere eine Fehlerbehandlung ganz nett
if [ ! -f "$File" ]; then
  echo "File $File not found/not readable"
  exit 1
  fi

if [ "$Subject" == "" ]; then
  Subject="Automatic Mail from Sysadmin"
  fi

# /etc/passwd zeilenweise bearbeiten
while read LINE
  do
  # Usernamen und Gruppennummer aus der Zeile fischen
  USR=`echo $LINE | cut -f1 -d:`
  GID=`echo $LINE | cut -f4 -d:`
  if [ $GID -eq 100 ]; then         # wenn die Gruppe passt
    echo "Sending $File to $USR"    # Mail verschicken
    mailx -s "$Subject" < $File
  fi
  done </etc/passwd
```

C.2 LaTeX

1. Schreiben Sie eine Muster-Präambel für einem Artikel, den Sie mit LaTeX verfassen wollen.

Lösung

```
\documentclass{article} % keine Kapitel
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[ngerman]{babel} % Neue Rechtschreibung
\usepackage{graphicx}
\begin{document} % Hier geht der Text los

\title{Ein toller Titel}
\author{Da stehe ich}
\maketitle
\newpage

\tableofcontents
\newpage
Hier geht dann der Text los
```

2. Erstellen Sie den LaTeX-Sourcecode für folgende Auflistung:

   Checkliste für mehrtägige Exkursionen

   - Waschzeug
     - Handtuch
     - Zahnbürste
     - Zahnpasta
     - Kamm/Bürste
     - Seife/Duschgel
   - Hausschuhe
   - Kleidung (genug)
   - Unterwäsche (ebenfalls genug)
   - Badehose
   - Trinkflasche
   - Schlafanzug
   - Ohrstöpsel
   - Wärmflasche (für Weicheier)
   - Notfallausrüstung
     - Handy
     - Verbandszeug
     - Alka Selzer
     - Rettungsdecke
     - MP3-Player
     - Taschenmesser

Lösung

```
\textbf{Checkliste für mehrtägige Exkursionen}

\begin{itemize}
  \item Waschzeug
    \begin{itemize}
     \item Handtuch
     \item Zahnbürste
     \item Zahnpasta
     \item Kamm/Bürste
     \item Seife/Duschgel
    \end{itemize}
  \item Hausschuhe
  \item Kleidung (genug)
  \item Unterwäsche (ebenfalls genug)
  \item Badehose
  \item Trinkflasche
  \item Schlafanzug
  \item Ohrstöpsel
  \item Wärmflasche (für Weicheier)
  \item Notfallausrüstung
    \begin{itemize}
     \item Handy
     \item Verbandszeug
     \item Alka Selzer
     \item Rettungsdecke
     \item MP3-Player
     \item Taschenmesser
    \end{itemize}
 \end{itemize}
```

3. Setzen Sie folgende Tabelle mit LaTeX:

Nahrungsmittel	Art	Ausgangsprodukt
Schnitzel	Fleisch	Schwein
Steak	Fleisch	Rind
Semmel	Getreide	Weizen
Pop Corn	Getreide	Mais
Joghurt	Milchprodukt	Kuhmilch
Knödel	Gemüse	Kartoffel

Lösung

```
\begin{tabular}{p{4cm} p{3cm} p{4cm} }
\hline
Nahrungsmittel & Art & Ausgangsprodukt \\
\hline
Schnitzel & Fleisch & Schwein \\
Steak & Fleisch & Rind \\
Semmel & Getreide & Weizen \\
Pop Corn & Getreide & Mais \\
Joghurt & Milchprodukt & Kuhmilch \\
Knödel & Gemüse & Kartoffel \\
\hline
\end{tabular}
```

4. Setzen Sie folgende Formel mit LaTeX:
   Mit $k = \sqrt[3]{8}$:

$$k^2 \in \left\{\sum_{i=1}^{\infty}\frac{1}{2^i}, \ldots, \sum_{i=1}^{\infty}\frac{1}{i}\right\}$$

Lösung

```
Mit \(k=\sqrt[3]{8}\):
\[ k^2\in\left\{\sum_{i=1}^{\infty}\frac{1}{2^i}, \ldots,
\sum_{i=1}^{\infty}\frac{1}{i}\right\} \]
```

5. Versuchen Sie, folgende kleine Mathelektion zu setzen:

   Jeder Mathematiker weiß, dass z. B. die Summe von zwei Größen nicht etwa in der Form

$$1 + 1 = 2 \tag{C.1}$$

   dargestellt wird. Diese Form ist viel zu schlicht. Schon Anfangssemester wissen, dass gilt:

$$1 = \ln e \tag{C.2}$$

   weiterhin ist geläufig, dass

$$1 = \sin^2 q + \cos^2 q \ . \tag{C.3}$$

   Ausserdem ist dem kundigen Leser offensichtlich, dass

$$2 = \sum_{n=0}^{\infty} \frac{1}{2^n} \ . \tag{C.4}$$

Daher kann die Gleichung (C.1) viel wissenschaftlicher in der Form

$$\ln e + (\sin^2 q + \cos^2 q) = \sum_{n=0}^{\infty} \frac{1}{2^n} \ . \tag{C.5}$$

ausgedrückt werden.

Lösung

```
Jeder Mathematiker weiß, dass z.\,B.\ die Summe von zwei Größen nicht
etwa in der Form
\begin{equation}\label{eq:simple}
1 + 1 = 2
\end{equation}
dargestellt wird. Diese Form ist viel zu schlicht. Schon Erstsemester
wissen, dass gilt:
\begin{equation}
1 = \ln e
\end{equation}
weiterhin ist geläufig, dass
\begin{equation}
1 = \sin^2 q + \cos^2 q \ .
\end{equation}
Ausserdem ist dem kundigen Leser offensichtlich, dass
\begin{equation}
2 = \sum^{\infty}_{n=0} \frac{1}{2^n} \ .
\end{equation}
Daher kann die Gleichung (\ref{eq:simple}) viel wissenschaftlicher
in der Form
\begin{equation}
\ln e + (\sin^2 q + \cos^2 q) = \sum^{\infty}_{n=0} \frac{1}{2^n} \ .
\end{equation}
ausgedrückt werden.
```

C.3 Programmieren mit Perl

1. Schreiben Sie ein Perl-Programm, das Zahlen einliest und sie aufsummiert. Wird statt einer Zahl der Buchstabe „e“ eingegeben, berechnet das Programm den Mittelwert der Zahlen und gibt ihn aus.

Lösung

```
use strict;

my $sum = 0;
my $count = 0;
# read first number
while (1)
  {
  print "Zahl eingeben: ";
  my $number = <>;
  chomp($number);
  last if ($number eq 'e');
  $count++;
  $sum += $number;
  }
print "$count Zahlen wurden eingegeben\n";
if ($count > 0)
  {
  print "Mittelwert: ",$sum/$count,"\n";
  }
```

2. Schreiben Sie ein Programm, das eine Zahl in Worten ausgibt (etwa für das Bedrucken eines Scheckformulars). Die Zahl soll dem Programm auf der Kommandozeile übergeben werden. Beispiel:

```
user$  perl drucke 1234
-eins-zwei-drei-vier-
user$
```

Verwenden Sie für die Zuordnung zwischen Ziffern und zugehörigem Zahlwort einen Hash.

Lösung

```
use strict;
my %numbers = (0 => "null", 1 => "eins", 2 => "zwei", 3 => "drei",
               4 => "vier", 5 => "fünf", 6 => "sechs", 7 => "sieben",
               8 => "acht", 9 => "neun");

my @characters = split (//, "@ARGV");
print "-";
foreach my $char (@characters)
  {
  $char = $numbers{$char} if ($char ge "0" && $char le "9");
  print "$char-";
  }
print "\n";
```

3. Schreiben Sie ein Programm, das Text von der Standardeingabe einliest und die Häufigkeit der einzelnen Buchstaben ermittelt. Die Liste der Häufigkeiten

soll dann alfabetisch ausgegeben werden. Hinweis: Eine Zeile in Buchstaben zerlegt man mit der Anweisung `@chars = split (//,$line);`.

Lösung

```
my ($char, $line, @chars);
my %freq = ();
while(defined($line = <>))
  {
  chomp $line;
  last if ($line eq '');
  @chars = split (//,$line);
  # Keine Unterscheid. gross/klein
  foreach $char (@chars)
    { $freq{lc($char)}++; }
  }

# Liste alphabetisch ausgeben
foreach $char (sort keys %freq)
  { print "$char: $freq{$char}\n"; }
```

4. Schreiben Sie ein Programm, das beliebig viele Zeilen Text von der Standardeingabe einliest und dien Text in umgekehrter Reihenfolge wieder ausgibt. Dabei sollen die Zeichen innerhalb der Zeile umgekehrt werden (`reverse`) und auch die letzte Zeile als erste ausgegeben werden (`push` und `pop`).

Lösung

```
my @lines = ();
my ($line, @chars);
while(defined($line = <>))
  {
  chomp $line;
  last if ($line eq '');
  # reverse the characters in the line
  # and push this onto a stack
  @chars = split (//,$line);
  @chars = reverse @chars;
  $line = join ("", @chars);
  push @lines, $line;
  }
print "$line\n" while ($line = pop @lines);
```

5. Erzeugen Sie eine Datei namens `gebtag`, die folgenden Text enthält:

```
11.2.1947:Thomas Alva:Edison
4.1.1643:Isaac:Newton
14.3.1879:Albert:Einstein
```

```
27.12.1571:Johannes:Kepler
```

Ergänzen Sie die Datei mit einem Testdatensatz, der das heutige Datum trägt.

Schreiben Sie nun ein Programm das nach der Püfung ob `gebtag` existiert, die Datei einliest und alle Personen heraussucht, die heute Geburtstag haben. Es sollen Vorname, Name und Geburtsjahr ausgegeben werden. Hinweis: Das aktuelle Datum erhalten Sie mit

```
($sek,$min,$std,$tag,$mon,$jahr) = localtime(time);
$mon++; $jahr += 1900;
```

Lösung

```
use strict;

my ($count, $datum, $vorname, $name, $tt, $mm, $jj);
my $datei = "gebtag";
my ($sek,$min,$std,$tag,$mon,$jahr) = localtime(time);
$mon++; $jahr += 1900;

print "Heute ist $tag.$mon.$jahr\n";

$count = 0;
open (GEB, $datei) || die "Kann $datei nicht öffnen: $!\n";
while (<GEB>)
  {
  chomp;
  # Kommentarzeilen erlauben
  next if (/^#/);
  # Zeile auftrennen in Datum Vorname Name
  ($datum,$vorname,$name) = split(/:/,$_);
  # Datum auftrennen in Tag, Monat, Jahr
  ($tt,$mm,$jj) = split(/\./,$datum);
  # Punkt ist spezielles Zeichen in reg. Ausdrücken, daher '\.'

  if ($tt == $tag && $mm == $mon)
    {
    print "$vorname $name hat heute Geburtstag ($jj).\n";
    $count++;
    }
  }
close GEB;
if ($count == 0)
  { print "Heute hat keiner Geburtstag.\n"; }
else
  { print "Heute haben $count Person(en) Geburtstag.\n"; }
```

6. Schreiben Sie ein Programm für den Administrator, das alle Prozesse eine auf der Kommandozeile angegebenen Benutzers killt. Rufen Sie dazu das `ps`-Kommando als Pipe auf und verarbeiten Sie dessen Ausgabe zeilenweise, indem Sie die Prozesse des angegebenen Users heraussuchen. Dann extrahieren Sie die Prozessnummer aus der Zeile. Perl kennt den Befehl `kill(Signal,Prozessnummer)` mit dessen Hilfe Sie die Prozesse löschen können. Führen Sie die Prozedur nacheinander mit den Signalen `TERM`, `HUP` und `KILL` aus.

Lösung (Hauptprogramm)

```
#!/usr/bin/perl
use strict;
# Kommandozeile fuer ps ggf. anpassen:
# Das Kommando liefert Prozessnummer und Username
my $pscommand = '/bin/ps -eo pid,user';

if ($#ARGV != 0)
  { print "Usage: $0 username\n"; exit(1); }

my ($signal, $error);
my $actuser = shift;

foreach $signal ("TERM","HUP","KILL")
  { &killuser($actuser, $signal); }
exit 0;

sub killuser # (user, signal)
  {
  # can only be done as root
  my $user = shift;
  my $sig = shift;
  my @entry = ();
  my $name = '';
  my $pid = 0;
  my $line = '';
  print "Sending $sig signal to all processes ...\n";
  # Prozessliste alle User einlesen, User extrahieren
  open(PS, "$pscommand |");
  @entry = grep(/$user/,<PS>);
  close PS;
  return if ($#entry == -1); # keine Prozesse da
  foreach $line (@entry)
    {
    chomp($line);
    # mehrere Leerzeichen durch eines ersetzen
    $line =~ s/  */ /g;
    # Leerzeichen am Anfang weg
    $line =~ s/^ *//g;
    ' Prozessnummer und Username extrahieren
    ($pid,$name) = split(/ /,$line);
    print "$pid ";
    kill($sig,$pid);
    }
  print "\n";
  return;
  }
```

Index

Symbole

A

E

M

N

O

P

Q

R

T